Z-200 Series

Technical Reference Manual

595-3522-2

TM-240

Zenith data systems

Z-200 Series

Technical Reference Manual

Zenith data systems

LIMITED RIGHTS LEGEND

Contractor is Zenith Data Systems Corporation of St. Joseph, Michigan 49085. The entire document is subject to Limited Rights data provisions.

RESTRICTED RIGHTS LEGEND

This computer software and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in the Governing Rights in Technical Data and Computer Software clause — subdivision (b)(3)(B) of DAR 7-104.9(a) (May 1981) or subdivision (b)(3)(ii) of DOD FAR Supp 252.227-7013 (May 1981). Contractor/ Manufacturer is Zenith Data Systems Corporation, of Hilltop Road, St. Joseph, MI 49085.

Copyright 1985, 1986 by Zenith Data Systems Corporation.
Printed in the United States of America

Zenith Data Systems Corporation
St. Joseph, Michigan 49085

Preface

The TM-240 Technical Reference manual is a base manual that provides technical information for the basic computer system. Technical data for optional equipment is available as appendices. These can be installed in the back of this manual to form an extension of the basic system hardware section, to allow you to customize this manual to your computer's specific hardware configuration.

Individuals with a lesser degree of experience can benefit through the logical progression from general to complex concepts in this manual. In contrast, an individual with a higher level of experience may wish to refer to the complex "System Programming" or "System Hardware" parts of the manual.

This manual focuses on the major assemblies and function programmability that comprise the computer system, and assumes the computer to be installed properly and operational. It also assumes that you can refer to the Z-200 PC Series Computer Owner's manual that was shipped with the computer. If you need assistance with installation and operation of the computer, refer to the owner's manual.

This manual is structured into four main parts. Chapters are contained within each part to allow the reader to conveniently refer to a given topic. A brief description of the information presented in each part of the manual is outlined in the following paragraphs.

Part I provides an introduction to the basic computer system. The assemblies and circuit cards that are installed in the basic computer are defined. A list of recommended publications also is included in this part of the manual.

The chapters in Part II provide a system overview that includes general theories of operation for the assemblies and circuit cards that comprise the basic computer system. Part discusses the function of these assemblies rather than complex circuit operation. In addition, this part of the manual defines the computer monitor program, self-tests, and error messages, and provides disassembly procedures and configuration data.

Part III is based on function programmability available to the computer. Setup program, video graphics, disk, keyboard, I/O ports, timer, real-time clock, system control, and specific monitor program data are provided here.

The chapters in Part IV analyze the complex circuits of the major computer assemblies that comprise the basic computer system. This text is supported with schematic illustrations, block diagrams, and discreet device pinouts.

The information presented in this manual will help you understand the true potential of the Z-200 PC Series Computer.

Contents

PART I INTRODUCTION

Chapter 1 **Introduction**

PART II TECHNICAL OVERVIEW

Chapter 2 **Hardware**

Power Supply 2-2
Backplane 2-2
CPU/RAM Card 2-4
I/O Card 2-5
Data Storage Controller Card 2-7
Floppy, Winchester, and Tape Backup Controller Card 2-7
Floppy and Tape Backup Controller Card 2-8
Data Encoding 2-8
FM Data Encoding 2-8
MFM Data Encoding 2-9
MMFM Data Encoding 2-10
Write Operations 2-10
Write Precompensation 2-14
Read Operation 2-14
Data Storage Devices 2-15
Floppy Disk Drives 2-15
Winchester Disk Drives 2-16
Tape Backup System 2-17
Video Interface 2-19
CRT/Monitor Operation 2-20
Keyboard 2-22
Peripherals 2-23

Chapter 3 **Firmware**

The Monitor Program 3-1
Self-Tests 3-3
Error Messages 3-4

Contents

Chapter 4 **Disassembly**

Z-248 Computer 4-1
Cover Removal 4-1
Card Removal and Replacement 4-2
Backplane Board Removal 4-4
Power Supply Removal 4-6
Floppy Disk Drive Removal 4-7
Drive A 4-7
Drive B 4-8
Winchester Disk Drive Removal 4-9
Drive 0 4-9
Drive 1 4-9
Cover Replacement 4-10
Tape Backup Unit 4-10

Chapter 5 **Configuration**

Setup Program 5-1
Time 5-2
Date 5-3
DST (Daylight Savings Time) 5-3
Base Memory 5-4
Expansion Memory 5-4
Floppy Drive 0 5-4
Floppy Drive 1 5-4
Boot Drive 5-4
Video Display 5-5
Video Refresh Rate 5-5
Winchester Disk Drives 5-6
Slow Mode 5-8
Z-100 Mode 5-8
Hardware Jumpers 5-9
CPU/RAM Card 5-9
EPROM Device Selection (J201) 5-9
Wait State Configuration (J202 – J206) 5-10
I/O Card 5-11
CRT Adapter Select (J301) 5-11
AC Voltage Present (J302) 5-12
Parallel Port Interrupt Request (J303) 5-12
Serial Port Interrupt Request (J304) 5-12
Parallel/Serial Port Address Select (J305) 5-12
High Density (1.2M) Floppy Disk Drive 5-13
Drive Select Jumpers 5-14
Motor Control Jumpers 5-16
Head Load Jumpers 5-17
Ready Status Jumpers 5-18
LED Indicator Status Jumper 5-19

Standard Density (360K) Floppy Disk Drive 5-20
Winchester Disk Drives (20M and 40M) 5-22

PART III SYSTEM PROGRAMMING

Chapter 6 **Software Interface**

MFM-200 Functions 6-1
MFM-200 Operation 6-2
Using an Interrupt 6-3
Modifying an Interrupt 6-5
Software Interrupt Summary 6-5
Hardware Interrupts 6-9

Chapter 7 **CPU Interrupts**

Programming CPU Interrupts 7-1
Divide by Zero 7-1
Single Step 7-2
Non-Maskable Interrupt 7-2
Software Breakpoint 7-2
Arithmetic Overflow 7-2
Timer (Time-of-Day) 7-3
Second 8259 Interrupt Controller 7-3
Equipment Configuration 7-3
Memory Size 7-5
Device Control Interrupt 7-5
Set/Read Time-of-Day 7-11
Timer Tick 7-12
Real-Time Clock 7-13
Software Redirect of IRQ2 7-13
80287 Non-Maskable Interrupt 7-13
Programming Sound 7-13
User Memory 7-14
Memory Address Format 7-15

Chapter 8 **Keyboard Interrupts**

Programming Keyboard Interrupts 8-1
Key Pressed 8-1
Input/Output 8-1
Keyboard Break 8-3

Chapter 9 **I/O Interrupts**

Programming Input/Output Interrupts 9-1
- Print Screen 9-1
- Communications 9-2
- Parallel Printer 9-2
- Serial Input/Output 9-2
- Printer Input/Output 9-6
- Parallel/Serial Configuration 9-7
  - Parallel Format 9-8
  - Serial Format 9-8

Chapter 10 **Data Storage Interrupts**

Programming Disk Drive Interrupts 10-1
- Floppy Disk Drives 10-1
- Disk Input/Output 10-1
  - Error Status Codes 10-10
- Booting an Operating System 10-11
- Disk Parameters 10-12
- Rigid Disk Drives 10-13

Chapter 11 **Video Interrupts**

Programming Video Interrupts 11-1
- Video Input/Output 11-2
- Video Initialization 11-11
- Graphics Characters 11-12

PART IV SYSTEM HARDWARE

Chapter 12 **Power Supply and Backplane**

Power Supply 12-1
- Specifications 12-1
- Line Potential 12-2
- Outputs 12-3

Backplane 12-6
- Specifications 12-6
- Bus Description 12-8

Chapter 13 **CPU/RAM Card**

Specifications 13-1
Central Processing Unit 13-2
Address Bus 13-2
Data Bus 13-4
Control Bus 13-6
The 80286 Instruction Cycle 13-7
Interrupt Requests 13-10
Maskable 13-10
Non-Maskable 13-10
CPU Control Logic 13-14
Fast Mode 13-15
Slow Mode 13-15
80287 Numeric Processor Extension 13-16
Random Access Memory 13-20
Memory Organization 13-20
Memory Address Bus 13-22
Memory Control Logic 13-22
Byte Data Access 13-25
Word Data Access 13-25
DRAM Refresh 13-25
Data Bus Interface 13-26

Chapter 14 **Input/Output Card**

Specifications 14-1
Address Decode Logic 14-2
Serial Port 14-3
Parallel Port 14-6
Port Decoder 14-8
Data Transfer 14-8
Writing to Port A 14-8
Reading Port A 14-9
Reading Port B 14-9
Writing to Port C 14-9
Reading Port C 14-10
Reset 14-10
Programmable Interrupt Controllers 14-10
Programmable Interval Timer 14-12
DMA Control 14-15
8-Bit DMA Control 14-17
16-Bit DMA Control 14-19
Keyboard Interface 14-21
Real-Time Clock Interface 14-22

Contents

Chapter 15 **Keyboard**

Specifications 15-1
Key Arrangement 15-4
Alphabetic Keys 15-4
Nonalphabetic Keys 15-6
Common Control Keys 15-8
Special Function Keys 15-9
Control Keys 15-11
Special Keys 15-12
Increasing Keyboard Buffer Size 15-13

Index

Appendices

Figures

1-1: Basic Computer System 1-1
1-2: Computer Backplane 1-2
1-3: Computer with Peripherals 1-5

2-1: System Block Diagram 2-1
2-2: Backplane LEDs 2-3
2-3: I/O Card LEDs 2-6
2-4: FM Data Encoding 2-9
2-5: MFM Encoding 2-9
2-6: Read and Write Signals 2-11
2-7: Write Operation Timing 2-13
2-8: CRT Signal Relationships 2-21
2-9: Keyboard Key View 2-22
2-10: Peripheral Equipment 2-23

4-1: Cover Removal 4-1
4-2: Card Support Brackets and Blank Brackets 4-3
4-3: Backplane Board Removal 4-5
4-4: Power Supply Removal 4-6
4-5: Typical Drive Bracket Assembly Removal 4-8
4-6: Cover Replacement 4-10

5-1: Setup Menu 5-2
5-2: CPU/RAM Card Hardware Configuration 5-11
5-3: I/O Card Hardware Configuration 5-13
5-4: Drive Select Jumper Location 5-15
5-5: Motor Control Jumper Location 5-16
5-6: Head Load jumper Location 5-17

5-7: Ready Status Jumper Location 5-18
5-8: LED Indicator Status Jumper Location 5-19
5-9: 360K Floppy Hardware Configuration 5-21
5-10: Winchester Configuration Jumpers (20M and 40M) 5-23

9-1: Serial and Parallel Device Layout 9-7

11-1: Character Design Matrix 11-13

12-1: Power Supply Block Diagram 12-1
12-2: Power Supply Output Connectors 12-5
12-3: Backplane Slots 12-7

13-1: CPU/RAM Card Block Diagram 13-1
13-2: Address Bus Schematic 13-3
13-3: Data Bus Schematic 13-5
13-4: Sequential vs Pipeline Processors 13-8
13-5: 80286 Device Pinout 13-13
13-6: 80287 Device Pinout 13-19
13-7: System Memory Map 13-21
13-8: Memory Address Bus Schematic 13-23

14-1: Serial Port Connector 14-4
14-2: Serial Port Controller Device Pinout 14-4
14-3: Parallel Port Connector 14-7
14-4: Programmable Interrupt Controller Device Pinout 14-11
14-5: Programmable Interval Timer Device Pinout 14-13
14-6: DMA Controller Device Pinout 14-17
14-7: Keyboard Interface Device Pinout 14-21
14-8: Real-Time Clock Chip Device Pinout 14-22

15-1: Alphabetic Keys 15-4
15-2: Nonalphabetic Keys 15-6
15-3: Common Control Keys 15-8
15-4: Special Function Keys 15-9
15-5: Control Keys 15-11
15-6: Keypad Special Keys 15-12

A-1: Backplane Signal Interface Schematic A-1
A-2: Control Bus Schematic A-2
A-3: CPU Control Logic Schematic, Part 1 .... A-3
A-4: CPU Control Logic Schematic, Part 2 .... A-4
A-5: Memory Control Logic Schematic A-5
A-6: I/O Card Block Diagram A-6
A-7: Address Decode Logic Schematic A-7

A-8: Serial Port Schematic A-8
A-9: Parallel Port Schematic A-9
A-10: Interrupt Control Schematic A-10
A-11: Programmable Interval Timer Schematic A-11
A-12: DMA Control Schematic, Part 1 .... A-12
A-13: DMA Control Schematic, Part 2 .... A-13
A-14: DMA Control Schematic, Part 3 .... A-14
A-15: DMA Control Schematic, Part 4 .... A-15
A-16: Keyboard Interface Schematic A-16
A-17: Real-Time Clock Interface Schematic A-17
A-18: Keyboard Schematic A-18

Tables

3-1: Error Messages 3-4

5-1: Z-200 Winchester Drive Types 5-6
5-2: Winchester Disk Drive Data 5-7
5-3: Wait State Configuration Jumpers 5-10

6-1: System Memory Use 6-2
6-2: System Port Map 6-3
6-3: Interrupt Summary 6-6
6-4: CPU Interrupts 6-7
6-5: Keyboard Interrupts 6-7
6-6: I/O Interrupts 6-8
6-7: Data Storage Interrupts 6-8
6-8: Video Interrupts 6-8
6-9: Hardware Generated Interrupt Requests 6-9

7-1: CPU Interrupts 7-1
7-2: Equipment Configuration (Register AX Report from INT 11H) 7-4
7-3: Interrupt 15H Functions 7-6
7-4: Joystick Values 7-7
7-5: Block Move Global Descriptor Tables 7-8
7-6: Global Descriptor Table Structure 7-8
7-7: Global Descriptor Tables for Function Code 89H 7-10
7-8: Device Type Codes 7-11

8-1: Keyboard Interrupts 8-1
8-2: Keyboard Status Report 8-2

9-1: I/O Interrupts 9-1
9-2: Serial Device Function Codes 9-2

9-3: Mode-Select Byte Breakdown 9-3
9-4: Word Length Selection 9-3
9-5: Parity Selection 9-3
9-6: Baud Rate Selection 9-4
9-7: Line Control Status (Register AH) 9-5
9-8: Modem Control Status (Register AL) 9-5
9-9: Parallel Device Function Codes 9-6
9-10: Parallel Map Format 9-8
9-11: Serial Map Format 9-8
9-12: Serial Byte #1 Breakdown 9-9
9-13: Serial Byte #2 Breakdown 9-9
9-14: Serial Byte #7 Breakdown 9-9

10-1: Data Storage Interrupts 10-1
10-2: Disk Drive Input/Output Ports 10-2
10-3: Drive Identification Codes 10-2
10-4: Disk Drive Function Codes 10-3
10-5: Drive Parameter Block 10-4
10-6: Required Parameters for Function Codes 02H – 05H 10-4
10-7: Register Requirements for Function Codes 06H, 07H, and 0CH . 10-5
10-8: Drive Type Characteristics 10-6
10-9: Register Requirements for Function Codes 0AH and 0BH 10-7
10-10: Disk Drive Error Codes 10-10
10-11: Disk Parameter Table 10-12
10-12: Sector and Gap Lengths 10-13

11-1: Video Interrupts 11-1
11-2: Video Input/Output Function Codes 11-2
11-3: Video Mode Codes 11-3
11-4: Video Initialization Default Values 11-12

12-1: Connector P1 Pinout 12-4
12-2: PC-Compatible Connector Pinout 12-9
12-3: PC AT Compatible Connector Pinout 12-11
12-4: Zenith Data Systems Connector Pinout 12-13

13-1: Bus Cycle Control Logic 13-9
13-2: 80286 Pinout 13-11
13-3: 80287 Pinout 13-17

14-1: Serial Connector Pinout 14-4
14-2: Parallel Connector Pinout 14-7
14-3: Parallel Port Selection 14-8
14-4: Programmable Interval Timer Programming Codes 14-15

15-1: Keyboard Processor Pinout 15-2
15-2: Alphabetic Key Codes 15-5
15-3: Nonalphabetic Key Codes 15-7
15-4: Common Control Key Codes 15-8
15-5: Special Function Key Codes 15-9

Listings

6-1: Sample Code 6-4

Part I

Introduction

Chapter 1
Introduction

The Z-200 PC Series Computers are professional business system computers that perform complex computing tasks. The computer serves as the base for a powerful system. Advanced state-of-the-art digital electronics and unique engineering concepts have been combined to form an exceptional and versatile computer. Refer to Figure 1-1.

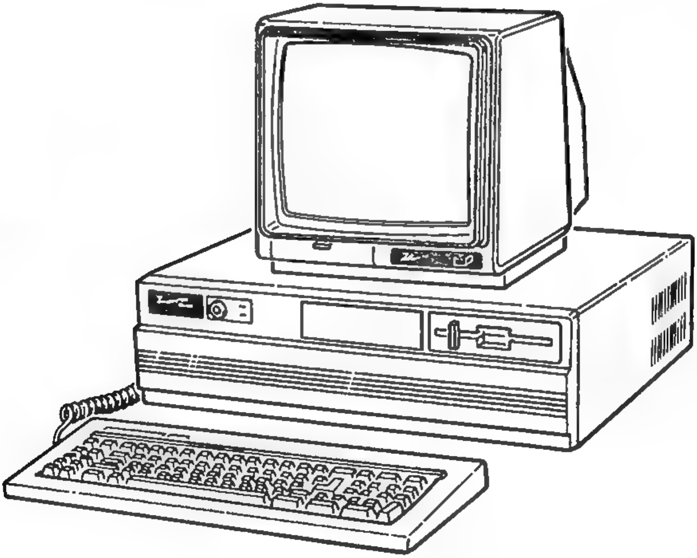


Figure 1-1: Basic Computer System

This high speed, multitasking, IBM PC AT compatible computer is based on the 16-bit 80286 microprocessor and 80287 numeric processor extension. This system runs under MS-DOS and XENIX operating systems. The basic computer is supplied with a parallel output port and a serial input/output port.

The basic computer incorporates a backplane, CPU/RAM card, I/O card, and some type of data storage controller card as follows:

- Backplane — The backplane has four slots that house two 62-pin connectors (Zenith slots), four slots that are IBM PC AT compatible and two slots that are IBM PC-compatible. Refer to Figure 1-2 for location of the numerically referenced backplane slots. The edge connectors in the slots accept the cards, and interface them to a 16-bit PC AT compatible bus and an 8-bit PC-compatible bus.

  Located on the backplane is a 5-pin DIN receptacle that the keyboard plugs into, and a battery backup circuit to power the computer's real-time clock. Six diagnostic LEDs that monitor the status of power supply operating voltages also are installed on the backplane.

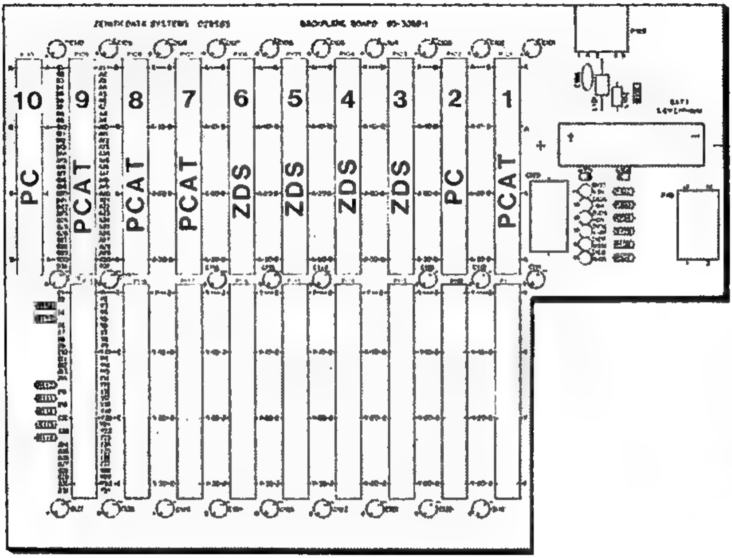


Figure 1-2: Computer Backplane

- CPU/RAM card — This card is located in slot number 3 (Figure 1-2) on the backplane. The CPU portion of the card uses the 16-bit 80286 microprocessor and supports the 80287 numeric processor extension. Logic required to generate 16-bit bus control signals, address and data buffering, and data directing is contained on the CPU portion of this card. The CPU runs with one wait state added to all memory cycles for PC AT speed compatibility.

  The RAM (random access memory) on this card consists of 512K with one parity bit for each 8-bit byte. The card contains on-board parity generation and checking circuits. There is 60K of ROM (read-only memory) available to the CPU and 4K of SRAM (special scratch-pad random access memory) used during ROM-based functions.

- Input/Output card — The I/O card is located in slot number 4, and contains the keyboard interface, speaker interface, serial port, parallel port, DMA control, interval timer, interrupt control circuitry, and real-time clock. This card also contains address decoding and control signal generation circuits that interface its address and data busses to the busses present on the computer backplane.

NOTE: Two data storage controller cards are available for use in the basic Z-248 computer. The actual card installed depends on the type of data storage device (floppy or Winchester drive) used in the computer. The following section briefly outlines the two types of cards available from Zenith Data Systems.

- Floppy, Winchester, and Tape Backup Controller card — This controller card is located in slot number 1 (in systems with one or more Winchester drive) on the backplane and is a custom manufactured, PC AT compatible circuit card. The controller circuitry supports up to two high performance Winchester disk drives, two floppy disk drives, and a tape backup unit. Two 360K standard density floppy drives, two 1.2M high density floppy drives (or one of each), two Winchester drives, and a tape backup unit can be interfaced to the computer through this card.

- Floppy and Tape Backup Controller card — This controller card is located in slot number 1 (in systems with one or more floppy drive, and no Winchester drives) on the backplane. The controller circuitry supports up to two floppy disk drives, and a tape backup unit. Two 360K standard density floppy drives, two 1.2M high density floppy drives (or one of each), and a tape backup unit can be used with the computer. This card may also contain an optional serial port.

The basic computer is not supplied with a video interface due to the popularity and availability of various after-market products. Therefore, a high degree of video interface flexibility can be achieved with the basic computer. The following video interface orderable options are available from Zenith Data Systems as follows:

- Extended Graphics card — The extended graphics card (slot 8) can support both color and monochrome video monitors, including the ZMM-1470 monochrome monitor and the ZVM-1380 enhanced color monitor. Features of this card include bit-mapped graphics and a RAM-loadable character generator. The extended graphics card contains five custom LSI devices that control signal generation and processing and 64K of RAM arranged as four 16K bit planes. Optional memory expansion can increase the RAM to 128K or 256K.

- Optional Z-409 Video card — The Z-409 color video card should be installed in a PC-compatible slot. This card supports monochrome, RGB, graphics (all points addressable), and text modes. Note that this card can be used in combination with the extended graphics card to achieve enhanced graphics.

- Optional Z-419 Bit-Mapped Video Graphics card — The optional bit-mapped video graphics card can be used in the Z-248 computer to provide Z-100 compatible graphics. This card must be installed in a PC-compatible slot. It provides an interface, via the extended graphics card, between the computer and monochrome or color monitors. This card provides high resolution (640 × 225), bit-mapped video output.

For many applications, peripherals can be added to the computer system. Some of these are illustrated in Figure 1-3. Please note that not all peripherals may be used at the same time. Refer to each peripheral's individual documentation for specific information regarding installation and configuration.

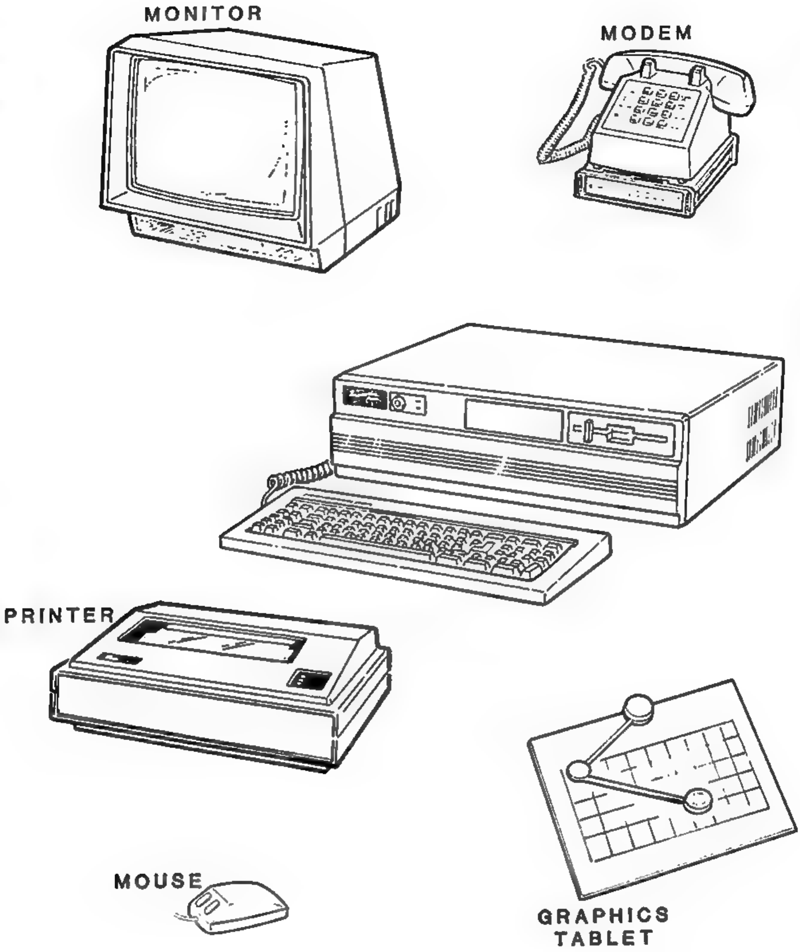


Figure 1-3: Computer with Peripherals

Because of the inherent complexity of this computer, we recommend the following publications to help you gain a more complete understanding of the computer.

- Z-200 PC Series Computer Owner's Manual — Describes the installation and operation of the computer in detail and includes a description of the hardware, its configuration, and basic maintenance.
- Intel iAPX 286 Hardware Reference Manual — This publication provides an in-depth study of the 80286 microprocessor and includes additional information pertaining to the 80287 numeric processor extension. This text is highly technical and provides an abundance of design criteria based on the 80286.
- Intel iAPX 286 Programmer's Reference Manual — This publication provides information on the architecture of the 80286 as well as the "basic" and "extended" instruction sets. Also included is information pertaining to CPU addressing modes, memory management, and system control.

The iAPX 286 hardware and programmer's reference manuals are published by Intel Corporation, 3065 Bowers Ave., Santa Clara, CA 95051

Part II

Technical Overview

The chapters in this part of the manual outline the abilities of the computer from a general technical viewpoint. Theories of operation of various hardware assemblies are provided as well as firmware, disassembly, and configuration data to help you develop a solid understanding of the computer at system level. In-depth studies of actual circuits are presented in Part IV.

Chapter 2
Hardware

The following section presents a hardware overview of the major assemblies in the basic Z-248 computer. A system block diagram of the computer is illustrated in Figure 2-1. Refer to this block diagram during the following discussions in this Chapter.

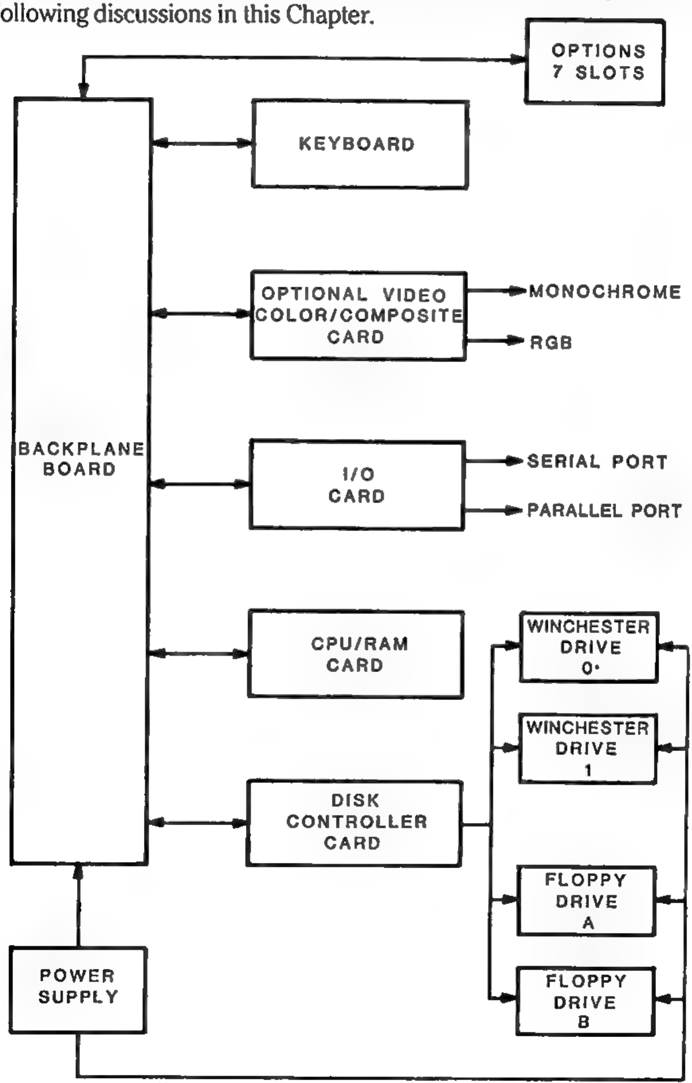


Figure 2-1: System Block Diagram

Power Supply

WARNING: The power supplies used in the Z-200 PC Series Computers do not contain user serviceable components. Because of the presence of very high DC voltages, do not remove the power supply covers under any circumstances. **Potentially lethal voltage** is still present even after the supply is disconnected from an AC source.

The power supply is a switching mode supply that generates a quasi-square wave signal. Filtered line voltage (approximately 170 VDC) acts as input potential to the switching elements. When these elements (field-effect transistors) are active, they chop the input voltage at a very high frequency that results in the quasi-square wave output. This resulting high magnitude pulsing DC signal is distributed across a transformer (toroid) to derive required output voltage levels. Each voltage is filtered and regulated. Switching mode supplies are highly efficient.

The power supply is located at the back right portion of the computer and provides power to the system. In addition, the supply provides operating voltage required to support two internal floppy disk drives and two internal Winchester disk drives.

Backplane

The backplane accepts the system cards, and interfaces data transmission between computer and keyboard. A 5-pin DIN connector is mounted on the backplane to allow the keyboard to interface to the computer.

The backplane accepts IBM PC AT compatible cards. Four slots with the appropriate edge connectors are supplied for this purpose. In addition, two slots are equipped with IBM PC-compatible connectors. This allows for a high degree of special performance flexibility due to the popularity of PC-compatible cards. Because Zenith Data Systems cards employ some special purpose signals throughout the system, the remaining four slots on the backplane are equipped with two 62-pin connectors in each slot.

There are six diagnostic LEDs on the backplane that monitor the status of power supply output voltages. All of the LEDs should remain lit during normal operation. If an LED fails to light, the output voltage of the power supply that drives the LED could be defective. Note that a defective current limit resistor, LED, or circuit trace could cause the LED not to light. Refer to Figure 2-2.

A lithium battery also is on the backplane to provide power to the real-time clock circuit when the main power is off. When you replace this battery, use a 3.6-volt, size AA, lithium battery.

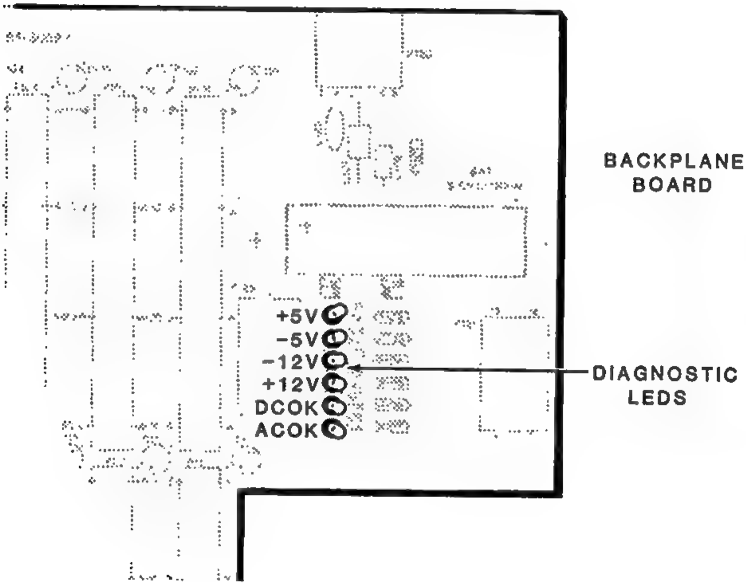


Figure 2-2: Backplane LEDs

CPU/RAM Card

This card employs the 80286 16-bit microprocessor. This powerful LSI (large-scale integration) device generates precise timing and control signals so that particular operations may occur. For example, to read data from DRAM (dynamic random access memory) the 80286 must assert a valid address. The 80286 then must assert the proper control and command signals to enable the DRAM devices, latch the address onto the DRAM_ADDR (DRAM address) bus, and enable particular buffers and latches residing on various data busses to transfer the data from memory to the 80286.

The microprocessor continually implements and monitors operations throughout the computer system. This is done through the address, data, and control busses. The address bus is used to initiate communication with many areas of the computer including memory, I/O circuits, and other cards present in the computer system. Byte or word data segments corresponding to a particular memory location or device can be placed on this bus by the 80286.

The control bus works in conjunction with the address bus. Status signals asserted onto the control bus enable particular areas (circuits) of the computer. The address bus then defines a specific location within a given area that the 80286 will communicate with. The 80286 uses the control bus exclusively during the generation of critical timing and control sequences and also uses it to determine the status of various devices residing on the control bus.

When the address and control busses enable a specific portion of the computer, perhaps a register within a device or a memory location within DRAM or ROM (read-only memory), the data bus transfers data between the CPU and the selected device. The control bus enables various bidirectional latches present on the data bus to direct data between device and CPU. The microprocessor in conjunction with various logic arrays generate critical timing signals to ensure that bus contention does not occur.

The DRAM on this card stores instructions and data that is used by the 80286 during its program execution. The basic computer configuration contains a minimum of 512K of memory, expandable to over 15M with the installation of optional cards. During the 80286 program execution the address, control, and data busses are used exclusively to allow the CPU to access DRAM.

In all cases, data contained in DRAM locations must receive refresh pulses at least every 4 ms to retain data integrity. If these pulses were not present, the logic levels of the data in DRAM would diminish rapidly. A binary counter generates addresses that enable entire memory location rows within DRAM. When a refresh cycle occurs, the data contained within the entire row gets refreshed.

The ROM contained on this card is addressed and read in the same way as for DRAM. The ROM of course, cannot be written to by the CPU. Particular instructions (firmware) that govern parameters such as powerup self tests, and error messages are contained within ROM. The basic system contains 60K of ROM and 4K of SRAM (special scratchpad RAM) for use during ROM-based functions.

I/O Card

Peripheral devices (parallel and serial) communicate with the computer through this card. The 80286 asserts bits onto the system address bus that are applied to a logic array address decoder. The logic array in conjunction with other gates, buffers, and multiplexers decode the I/O card parallel and serial port addresses.

The serial port acts as an interface to conduct communication signals between the computer system busses and a peripheral (serial) device. The parallel port interfaces parallel data between the computer and a peripheral (parallel) device. The parallel port actually consists of three individual ports. One port (B) is used as a read-only port to interface device signals to the parallel data bus. The other two ports (A and C) are read and write ports used to interface data and control signals between the computer and the peripheral device.

When a peripheral device (serial or parallel) is active, it triggers an interrupt request that causes the CPU to devote its program execution to the active port. It is necessary for the CPU to recognize that an operation is taking place at the port so that data is not inadvertently lost or misdirected. Many times the CPU will instruct a DMA (direct memory access) controller to implement data transfers between computer memory and peripheral devices. The DMA controller can transfer data much faster than the CPU. The DMA controller takes control of the system busses during a data transfer so no other devices can occupy the busses at this time.

In addition to interfacing signals to peripheral devices, the I/O card also interfaces data transmission to and from the keyboard. A system control processor transforms serial transmission from the keyboard into bytes that the 80286 can process. An interrupt request is generated when the keyboard transmits data. The interrupt request draws the attention of the 80286 to the system control processor.

A real-time clock chip with RAM on the I/O card allows the CPU to keep the correct time and date. This device receives power from a battery backup circuit when the computer is not in use. The battery backup allows the real-time clock chip to retain the system configuration information in the memory section, and to keep the clock/calendar running when the main power is off.

The I/O card also contains six diagnostic LEDs. These LEDs provide a visual indication of the self-tests that execute at power-up. All of the LEDs should light and then go out sequentially. The RDY LED will go out when an operating system is read into computer memory. If any LEDs remain lit, it indicates that a hardware failure (of circuits the LED corresponds to) has been detected. Refer to Figure 2-3.

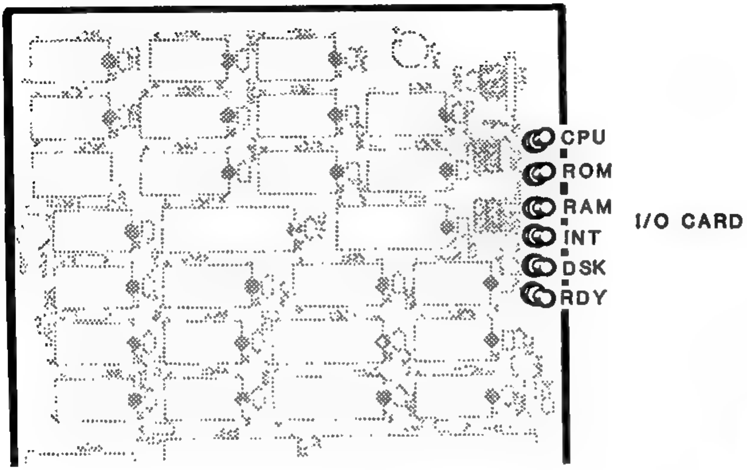


Figure 2-3: I/O Card LEDs

Data Storage Controller Card

There are various cards that can be used in the Z-200 PC Series computers to control the transfer of data to and from a given storage device. This section presents a brief description of two types of controller cards. Noted that the actual card installed in the computer is dependent on the type of mass storage devices used in the computer. The appropriate appendix in the back of this manual supplies more complete technical information (including configuration data) for the data storage controller card installed in your computer.

Floppy, Winchester, and Tape Backup Controller Card

This controller card connects to the PC AT compatible slot nearest the drive assemblies (slot 1). The card interfaces data to the selected storage device (floppy, Winchester, or tape backup). The maximum configuration allows for two floppy disk drives, two Winchester disk drives, and one external tape backup unit.

Data is transferred between controller and either Winchester disk drive at a 5-megabyte/second rate. Data transfer is performed in 16-bit words with a maximum transfer rate of 400 ns per word. The data transfer rate for either floppy disk drive and tape backup unit is software selectable on a per operation basis. Data transfer can be performed in 8-bit bytes by the CPU or through DMA (direct memory access) control.

Some features of the controller card include:

- On-board diagnostics.
- Supports disk drives with fast access step methods.
- Supports up to two Winchester drives, two floppy drives, and one external tape backup unit.
- Supports FM and MFM data encoding.
- Allows primary and secondary drive addressing.

Floppy and Tape Backup Controller Card

This card supports two internal floppy drives and provides an external connector to interface control to external floppies or a tape backup unit that is compatible with the floppy disk format.

The card occupies one standard PC compatible slot. A 34-pin connector is mounted on the card that interfaces control to internal floppy drives. A 37-pin connector is mounted in the card bracket that interfaces control to external floppy drives or a tape backup unit.

An optional serial port may also be installed on this card. Data transmission is interfaced through a 10-pin connector and a cable that runs over to the access panel on the rear of the computer. The access panel has a 9-pin connector mounted in it to allow connection to a peripheral device that recognizes serial data transmission. A DIP (dual inline package) switch is provided to select between primary and secondary address space for the serial port (refer to Chapter 5, "Configuration"). The serial port also can be totally disabled by a switch.

Data Encoding

To properly record data on magnetic media, the data is grouped into one of three different recording formats, although the Z-200 PC Series computers use only two of these formats. Data bits can be encoded and recorded as 8-bit bytes, a number of bytes within a sector, a number of sectors within a track, and various tracks on both sides of a disk.

FM Data Encoding

FM (frequency modulation) encoding is the simplest form of encoding that positions dipoles (magnetic particles in the disk structure) to be interpreted as data bits. It provides for an easy means of decoding because each data pulse occurs between two consecutive clock pulses that create a very precise "Read Window." Refer to Figure 2-4.

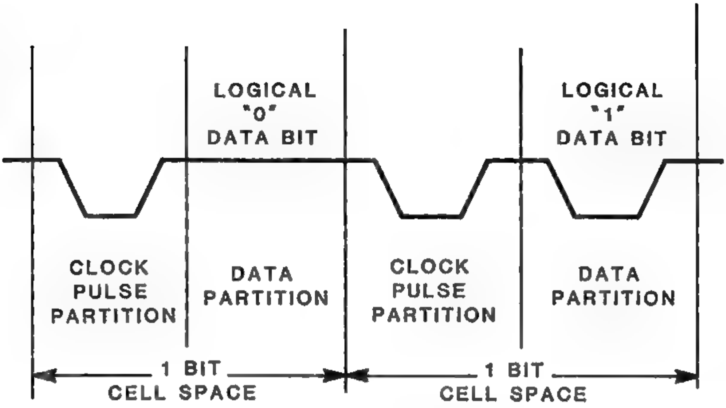


Figure 2-4: FM Data Encoding

MFM Data Encoding

MFM (modified frequency modulation) encoding aligns twice as many dipoles as FM encoding does. Data bits are deposited within the same linear recording density (one inch) in either format. In MFM encoding format, clock pulses are removed from the write data input to allow the bit cell space to be reduced by 50 percent. Refer to Figure 2-5.

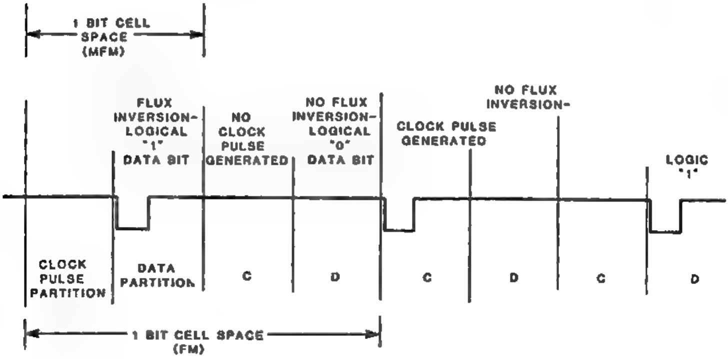


Figure 2-5: MFM Encoding

Each bit cell space contains a clock and data bit partition. Data bits are written at the leading edge of the data partition of the bit cell space. The presence of a high-to-low transition causes flux inversion on the media structure to be recognizable as a logical 1. The absence of a high-to-low transition within the data bit partition does not cause a flux inversion. The media structure does not change for the given bit cell space, making it recognizable as a logical 0.

If a data pulse is not present within two consecutive bit cell spaces, a synchronization clock pulse will be generated within the clock bit partition of the second bit cell space. The synchronization pulse is modified before application to the bit cell space so that it accesses only the clock partition of the cell space.

MMFM Data Encoding

MMFM (Modified-Modified Frequency Modulation) encoding, not illustrated, is a nonstandard double-density encoding method. Fewer clock pulses are used with respect to MFM encoding to provide synchronization between data bit pulses. The presence of a high-to-low transition at the leading edge of the data partition of the bit cell space causes flux inversion on the media structure, that is recognizable as a logical 1. The absence of a high-to-low transition at the leading edge of the data partition may be interpreted as a logical 0. A clock pulse is encoded only if no clock or data pulses were present in the previous bit cell space, and a data pulse is not to occur in the present bit cell space.

Write Operations

Recording on the disk is similar to recording audio or video signals on tape. Briefly, two frequencies are used to record FM encoded digital signals on the disk: 62.5 kHz and 125 kHz. The higher frequency is used to record adjacent (clock and data bit) pulses while the lower frequency records non-adjacent (clock only) pulses. MFM encoding modifies the data and clock-to-frequency relationships, but the recording and playback methods are identical.

Cores are present in read/write heads and erase heads that have coils wound at some point on the rings. When current flows through the coil of a read/write head, a magnetic field develops at the gap in the ring (refer to Figure 2-6). As the disk's surface passes the gap, the magnetic field magnetizes the recording surface in a longitudinal manner. When the current flow through the coil is forced to change its direction, the magnetic structure of the disk changes to represent a clock pulse or data bit (depending on the type of encoding used).

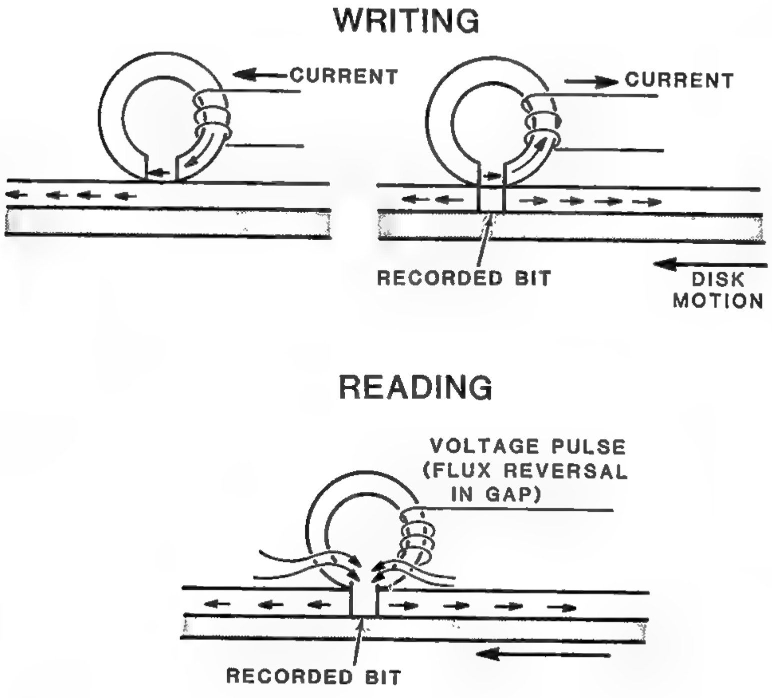


Figure 2-6: Read and Write Signals

The erase cores are energized during a write operation. The current flow through the erase coils remains stable, causing the surface of the disk passing under it to be erased. These two cores (located on either side of the read/write core) "trim erase" the track being recorded so that it will not exceed 0.012 inches (48-tpi recording) or 0.006 inches (96-tpi recording). This "tunnel erasing" prevents minor deviations in the recording current (passing through the read/write head) from affecting the magnetic structure of adjacent tracks of the disk.

During a write operation, a data bit is recorded on the disk when the magnetic field produced by the ring (in the read/write head) is reversed by rapidly reversing the current through the coil around the ring. This "flux" reversal is almost instantaneous in comparison to the motion of the disk, so the portion of the disk that just passed under the gap is magnetized in one direction while the portion currently under the gap is magnetized in the opposite direction. The point of flux reversal on the disk represents a data bit.

During a read operation, the rotation of the disk produces a constant magnetic field at the gap in the core of the read/write heads. The core senses the polarity of the magnetic field produced by the surface of the disk. When the magnetized surface of the disk (where a data bit exists) causes the magnetic field at the gap of the ring to change abruptly, a voltage (and current) is induced into the coil around the ring. The electronic circuits in the disk drive then convert this voltage into the clock or data pulse that the magnetic reversal on the disk represents.

The method of recording and playing back the pulses may vary from manufacturer to manufacturer, but the principles remain the same. The result is that recorded signals produced by one disk drive may be read by any other disk drive that uses the same principles.

At the beginning of a write cycle the Write Gate signal is driven low. The write driver circuitry then is enabled to transmit MFM encoded data. Data is transmitted over the "Write Data" signal line to be recorded onto the selected media. When the state of the Write Gate signal is high, the write driver circuitry is disabled to allow the "read data" logic circuits to activate. Refer to Figure 2-7 for additional timing information.

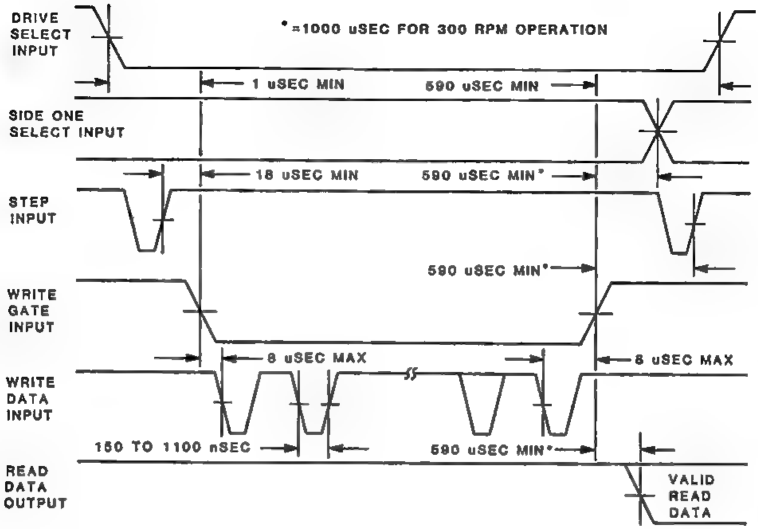


Figure 2-7: Write Operation Timing

Data transmitted to the storage device is conducted over the Write Data signal line. When data is not being written, the state of this line is high. As a Write Data pulse occurs, the high-to-low transition of the pulse causes current reversal through the enabled write head. The polarity of the current flow through the head creates a flux inversion that aligns dipoles in the media structure accordingly. The arrangement of the dipoles with respect to polarity are interpreted as data bits.

Write Precompensation

As data is deposited onto the surface of a disk nearer and nearer to the center of the disk, the space between data bits effectively becomes less and less. Recalling that a logical data bit is actually a dipole in the disk structure that has been positioned with reference to polarity, it is logical to assume that a fixed magnetic field surrounds each respective dipole.

If data bits were written onto the disk at a constant rate, the bits (dipoles) would effectively become closer and closer to each other. If this were allowed to occur, the magnetic fields surrounding one dipole (bit) would begin to intersect the magnetic fields of other adjacent dipoles (bits). Since oppositely charged or polarized particles tend to attract each other, and like charges or polarities repel, a phenomena referred to as "bit shift" can result when two bits (dipoles) are deposited too closely to each other.

To prevent bit shift from occurring, electronic precompensation timing is used to make sure that a required minimum amount of space, proportional to the rate at which available disk structure decreases, exists between data bits (dipoles). The effect that bit shifting has on adjacent dipoles can be anticipated and therefore corrected. If precompensation were not used, data integrity near the center of the disk would be considerably less than near the outer edge of the disk.

Read Operation

Data bits are read from the media structure by the read heads. When the heads detect a flux inversion in the disk structure (logical 1) the "read data" line is asserted low. The floppy controller portion of the controller card interprets the low pulse as a logical 1.

Data Storage Devices

Floppy Disk Drives

The floppy disk drives used in the Z-200 PC Series Computers can be either standard density (360K) or high density (1.2M) disk drives. The high density drives are double-sided, double-density drives that have an unformatted storage potential of 1.6M (360 rpm rotation).

A brushless DC motor provides disk rotation when the drive is active. The motor in the high density drive is capable of rotating the disk at 360 rpm. When the disk is rotated at 360 rpm, the drive assembly is capable of transferring information to and from the disk at a rate of 500 kilobits/second. The motor in the low density drives rotates the disk at a slower rate (300 rpm). The rate of transfer between disk and interface is reduced to 250 kilobits/second in the standard density drive.

When the drive is initially powered up, the read/write heads may be arbitrarily positioned from previous use. To bring them to a known position (track 00) the drive automatically will step the heads across the media (disk) until the track 00 signal is detected. The movement of the heads will then be stopped.

The assembly that supports the read/write heads is fastened to a positioning mechanism that is driven by a stepping motor. During program execution the stepping motor moves the heads the required number of tracks in either of two designated directions. The status of the direction signal determines which way the heads will be moved, and the active step signal causes the stepping motor to move the heads in that direction. Step pulses are provided until the head reaches the desired track.

There are read/write heads located on both sides of the disk. The drive assembly is therefore capable of enabling either set of heads to read from or write to either side of the disk. The status of the "side 1" select signal determines which set of heads to enable. If the side 1 select signal is high, the read/write heads contacting side 0 (of the disk) will activate. If the side 1 select signal is low, the read/write heads contacting side 1 (of the disk) will activate.

Data is transferred to and from the determined side of the disk through the read/write heads. These heads are designed with three ferrite cores. Two cores, one on each side of the read/write core, are designated as erase cores. They are used to erase the area between tracks on the magnetic disk. The heads themselves are positioned facing each other. The disk of course, travels within the area between each set of heads. When a disk is installed into the drive and the latch is closed, the heads are brought into contact with the disk. Specific mechanisms make sure that the heads track the disk very consistently so that data is accurately reproduced. Data is transferred to and from magnetic disks through either FM, MFM, or MMFM encoding. FM encoding is used during single-density data transfer while MFM and MMFM encoding is used during double-density data transfer.

Winchester Disk Drives

The Winchester disk drives used in the Z-200 PC Series Computers are very high performance drives capable of storing as much as 20M or 40M of data (depending on the drive purchased). All necessary mechanical and electrical parts to interpret control signals, position the recording heads over the desired track, read and write data, and provide a contaminant-free environment for the heads and disks are self-contained.

A brushless DC motor rotates the hard disks at 3600 rpm. This assembly is dynamically balanced to provide minimal mechanical runout to the disks. Disk rotation is continually monitored by an internal microprocessor to make sure that the speed (3600 rpm) does not vary. The motor that drives the disks is powered by + 12 VDC.

When power is initially applied to the Winchester disk drive, the microprocessor verifies that the disks are rotating at 3600 rpm and then automatically activates the “track 000” positioning sequence. This is necessary so that the read/write heads are positioned at a known track on the disk. To bring the heads to this known position (track 000) the drive automatically will step the heads across the media (disk) until the track 000 signal is detected. The internal track 000 detector circuit consists of a light source, that is activated during the “seek” operation, and a receiver. When the light source is interrupted by the positioner assembly of the drive, the microprocessor interprets it as track 000.

The read/write heads are supported by an assembly that is coupled to a stepper motor through a rack-and-pinion motion translator. This allows for the increased number of data tracks while retaining the full step-holding torque and positioning repeatability characteristics of the stepper motor. Stepping pulses drive the stepping motor that in turn moves the read/write heads. Each step pulse causes the heads to be moved one track in or one track out, depending on the status of the direction signal. A true direction signal will cause the heads to move toward the center of the disk. A false direction signal causes the heads to move toward track 000. The drive prevents any outward movement beyond track 000 regardless of the number of step pulses.

Data is recorded on lubricated, magnetic oxide-coated aluminum substrate (disks) through low-force, low-mass Winchester-type ferrite heads. A valid binary address, placed on the head select lines of the drive, can select any of the internal heads. Each head has its own specific address so that two or more heads are not active at the same time.

When a particular operation such as a "read" operation is active, the write circuitry is disabled. In addition, the heads in the drive must be waiting at track 000 and the appropriate head-select binary address must be asserted. Data can then be transferred between the disk and controller.

When data is to be written onto the disk, the "write gate" signal must be true, which also disables the read circuits. Data must then be encoded in MFM format and transferred to the disk over the "MFM write data lines."

Tape Backup System

The following section describes one type of external 20-Megabyte tape backup system. This tape backup system is a high-capacity streaming tape system for use with high-capacity fixed-disk storage devices. The external system is attached to the host computer and controlled by special software. The performance of this system is similar to the industry standard streaming tape drive.

Software includes tape format, backup, and restore functions that are similar to the FORMAT, BACKUP, and RESTORE functions of MS-DOS. These include file-oriented storage to back up or restore a specific file, directory or disk, although the system offers the most effective transfer of data by streaming. Finally, the system provides protection against accidental overwriting of existing data and out-of-sequence cartridge loading.

The computer interfaces with the tape backup system through a 34-connector cable that carries both data and control signals. The control signals received from the computer are interpreted by the tape backup system to initiate the appropriate function.

The tape backup system has an on-board microprocessor that directly controls the operation of the drive by executing a control program stored in on-board ROM. Firmware routines are also provided to:

- Interpret motion, reporting, and mode commands transmitted by the host computer using the STEP signal.
- Control the speed and direction of the drive motor.
- Position the read/write head.
- Permit the host computer to record formatted data in assigned data areas.
- Record servo patterns on blank tape.

A brushless DC drive motor is mounted on the underside of the frame. The motor's drive shaft projects up through a bearing in the main frame and turns a rubber drive capstan. When a tape cartridge is installed, this drive capstan is pressed against a wheel inside the tape cartridge to move the tape bidirectionally against the read/write head.

The positioner mechanism for the read/write head moves the head across the width of the tape. The head itself is moved by a spiral cam that is driven by the positioner motor. The rotation of this cam raises and lowers an arm that is mechanically linked to the read/write head. By half-stepping the positioner motor, the head is moved in small linear increments, which coarsely positions it over one of the eight narrow tracks. Fine positioning is then achieved by reading prerecorded servo data from the tape itself and using this data to readjust the positioner motor until the read/write head is centered on the track.

The tape backup system also contains three sensors. The EOT/BOT (end-of-tape and beginning-of-tape) sensor is an infrared emitter whose light is projected through holes in either end of the tape. These holes allow the infrared light to strike the sensor's receiver to identify the end and beginning of the tape.

The WRITE PROTECT and CARTRIDGE PRESENT switches are subminiature switches whose status is detectable by the tape backup system's firmware. The CARTRIDGE PRESENT switch is engaged any time a cartridge is installed in the unit. The cartridge can be write-protected by the moving the cartridge's RECORD slide to the right, which disengages the WRITE PROTECT switch. Moving the slide to the left enables writing to the cartridge. A write-protect status signal is also sent to the floppy disk controller card.

Video Interface

Note that the video interface for the Z-200 PC Series computers is considered optional. Many different types of video interfaces can be used in the computer. Consult the specific documentation for the particular video interface that you intend to use, for installation and configuration information.

The video interface provides communication between the system busses and a peripheral monochrome or color monitor. The 6845 CRT (cathode-ray tube) controller is a popular LSI device that derives the proper signals to generate a raster on a CRT. The CRT controller is a highly programmable device with regard to raster and character manipulation. The programming capabilities of the CRT controller allow many different modes of operation including text modes and variations of graphics modes.

Mode select and status registers of the video interface determine the desired mode of operation and whether color or monochrome display will be generated on a CRT. These registers are usually programmable I/O registers.

Data that is transferred between CRT controller, CPU, and peripheral monitor generally pass through a common element referred to as a display buffer. Display buffers usually are DRAM devices with a dual-port arrangement to allow both the CPU and CRT controller access to it during any mode of operation.

Most video interfaces allow for text modes and variations of graphics modes. During the text mode of operation, a character ROM generates characters from dot patterns contained within the ROM's firmware. Different character fonts are available through the various video interfaces; a 7 × 7 double-dot and a 5 × 7 single-dot width character set are two popular fonts.

A color video encoder usually accepts information from the character ROM and mode select registers along with appropriate timing and control signals, then generates color video "dot" information. This information is sent to video output circuitry that routes the video information to the proper output connector on the video interface being used. In general, RGB color information is sent to an industry standard 9-pin connector, and monochrome composite information is sent to an industry standard RCA phono-jack type connector.

Many special performance video interface cards also are available to provide high resolution (typically 640 × 225 or greater), bit-mapped video output to the video monitor.

CRT/Monitor Operation

The external monitor used with the Z-200 PC Series computers must be a raster-scan device. Raster-scan CRTs are most commonly used in microcomputer applications.

The electronic circuitry in a monitor supplies the necessary operating voltages to generate a beam of electrons against phosphors that are deposited on the inside face of the CRT. This agitates (or excites) the phosphors, producing visible light. In order to produce a recognizable image, individual points called pixels (picture elements) must be lit through this process.

The entire displayable portion of the screen is scanned by the beam of electrons forming the images you see. CRT scanning takes place in a left-to-right, top-to-bottom motion (as you face the CRT). Since the beam must be returned to the left and top of the CRT's screen, it is turned off during this retrace time (blanked). Figure 2-8 illustrates the timing of the composite signal that produces the image you see on the face of the CRT. The relationships illustrated are not absolute but roughly represent the actual signals.

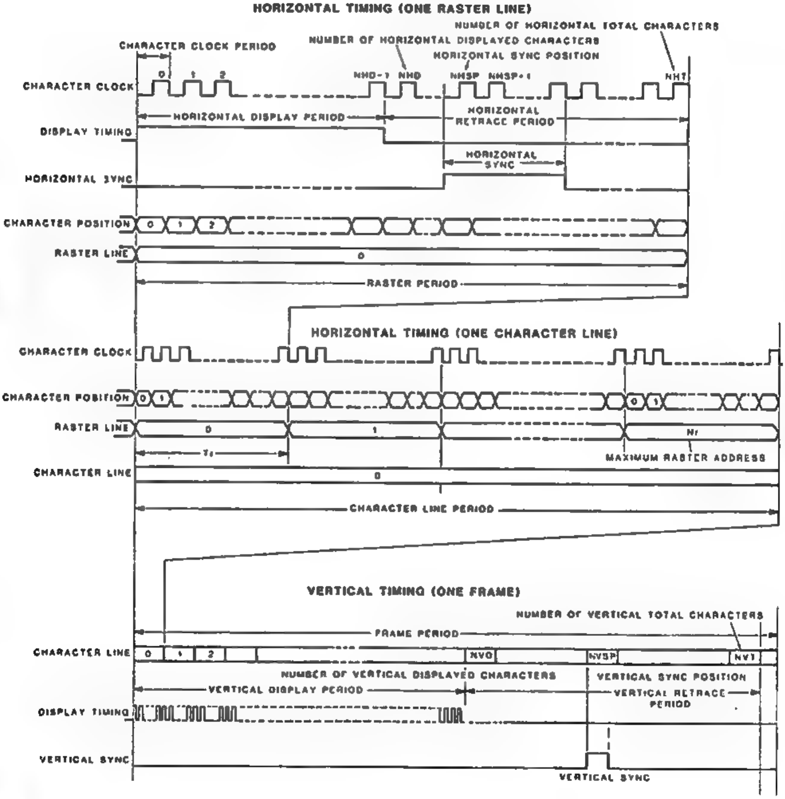


Figure 2-8: CRT Signal Relationships

Keyboard

The keyboard contains 84 keys laid out in three major groups. The central portion of the keyboard consists of a standard (QWERTY) typewriter keyboard layout. On the left side, arranged as a 2 × 5 block, are 10 function keys. These keys can be user-defined. On the right side is a 16-key, keypad area. These keys are defined by software but contain legends for the functions of numeric entry, cursor control, and calculator pad screen edit. See Figure 2-9.

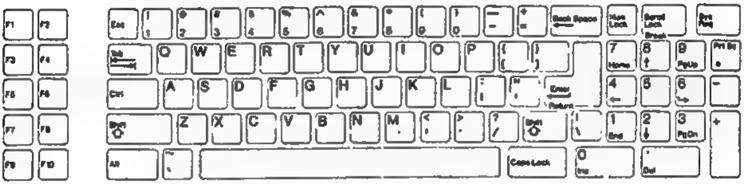


Figure 2-9: Keyboard Key View

The keyboard interface is defined so system software has the maximum flexibility in defining keyboard operation. This is accomplished by having the keyboard return specific hexadecimal codes rather than ASCII codes. In addition, all keys except control keys are event-driven and generate both “make” and “break” codes. For example, key 1 produces hexadecimal 01 when pressed and hexadecimal 81 when released. The keyboard I/O (input/output) driver can produce code either with or without control keys (SHIFT, CTRL, ALT) pressed, or under event-driven conditions as required by the application.

The keyboard cable is a coiled, shielded, 5-wire cable. The cable interface contains power (+5 VDC), ground, and two bidirectional signal lines. The cable is permanently attached at the keyboard and interfaces with the system control processor via a DIN connector located on the computer backplane.

The keyboard uses an 8049 microcontroller to perform keyboard scan functions. Additional keyboard functions include: key debounce, buffering of up to 16 key scan codes, maintaining bidirectional serial communications with the system control processor, and executing the handshaking protocol required by each code transfer.

Peripherals

For many applications peripheral equipment can be added to your computer system, such as a high-resolution color monitor, parallel printer, modem, graphics tablet, or a mouse. The industry standard connectors used with the computer are conveniently located on the back panel of the computer.

Please note that not all of the peripherals may be used at the same time. For example, if you have two peripherals that recognize parallel data signals, you can connect only one of them to the single parallel port on the computer. If you need to use the second peripheral, you must install a second parallel interface adapter, or disconnect the first one. Some popular peripherals are illustrated in Figure 2-10.

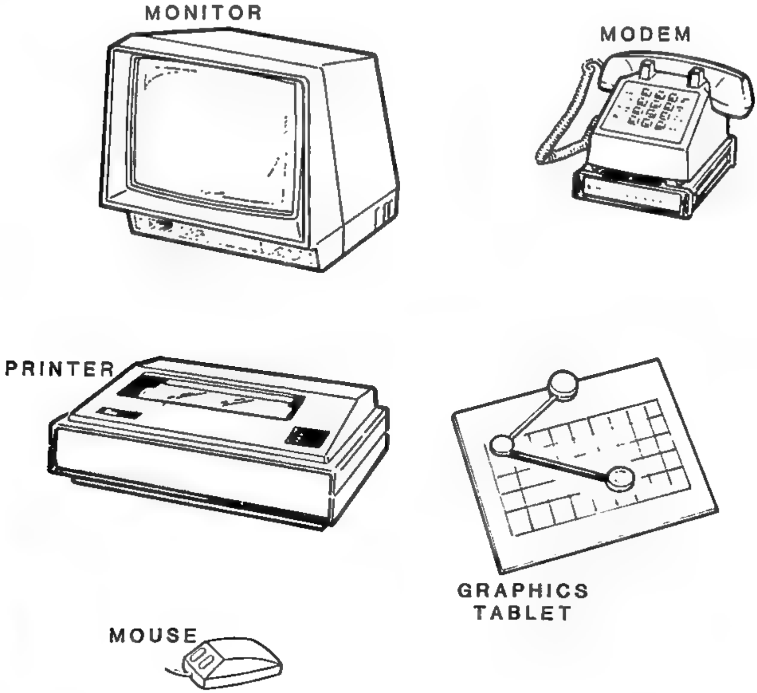


Figure 2-10: Peripheral Equipment

Chapter 3
Firmware

This chapter defines the monitor program, self-tests, and error messages generated by the computer.

The Monitor Program

The computer contains a special program called a monitor. One of its functions is to load (or boot) the DOS (disk operating system) from disk into memory. Computers are set at the factory to automatically boot the DOS when they are turned on. The autoboot feature can be disabled by changing the Setup program. Refer to Chapter 5, "Configuration" in this part of the manual for detailed information regarding the Setup program.

If the autoboot feature is disabled, a message similar to the following 3-line message will be displayed in the upper left-hand corner of the monitor's screen after powerup.

```
MFM-200 Monitor, Version 1.1
Memory Size: 512K
Enter "?" for help.
->
```

Professional software is released in versions. Updates and enhancements to these programs are made as they are developed. Each time a revision of this type occurs, it is called a version and given a number (Version 1.0, Version 1.1, etc.) The version of the monitor program is shown on the first line of the above 3-line message.

The second line of the above message indicates how many thousands (K) of bytes of memory are installed in the computer. In this case, the computer used has 512K of memory and firmware version 1.1 installed in it.

To execute any monitor program command, the prompt (->) must be present. It occurs automatically after the system is turned on and may be reset at any time by pressing the **CTRL**, **ALT**, and **INS** keys at the same time. Note that resetting the computer causes data contained within system RAM to be lost. If the autoboot feature is enabled, there will be no display until the DOS is read into system RAM. If no DOS is read, an eventual error message will be displayed.

Computers with Winchester drives may attempt to boot from the Winchester operating system. If the DOS has not been placed on the Winchester disk, an error message eventually will be displayed.

If your computer attempts to boot a disk from an empty floppy drive, the disk access light will remain on and the following message will be displayed after about 20 seconds:

```
+++ DISK ERROR: Drive not ready! +++
```

Pressing the **ESC** key will drive the computer to the monitor program. The opening messages will be displayed, erasing the error message. The computer is then in the manual mode and ready to execute monitor commands.

The B (boot) command is the monitor command most often used, particularly if your computer is not set to autoboot. The boot process reads the operating system from the disk and loads it into system RAM. To use this command:

- Make sure the computer and video monitor are on and the monitor prompt is showing.
- Insert the MS-DOS disk (or any disk with a Z-248 computer disk operating system on it) into drive A.
- Close the disk drive door or latch.
- Press the B key. The computer will display the letter B (or b).
- Press the **RETURN** key.

The computer will now execute the boot routine and load the operating system from the disk into the computer's memory.

Self-Tests

On powerup the computer executes a series of internal tests to check that everything is working correctly. These tests verify that all of the circuits are in a starting configuration and check various functions of the computer that allow it to operate properly. The following checks are made during the self-test sequence:

- Setup menu configuration
- Crystal frequencies
- Interrupt controllers
- DMA (direct memory access) controllers
- Disk drive controllers
- Disk drives (if enabled in setup)
- CPU (central processing unit)
- ROM (read-only memory)
- RAM (random access memory)

The video, keyboard, DMA, interrupt, disk, refresh, and any other controller in an expansion slot with an initialization ROM, will be initialized during the self-test sequence. When the tests are finished, the computer will display an opening message, or start the automatic boot procedure (autoboot). If autoboot to the floppy drive is started, a disk must be placed in the drive and the door shut within about 20 seconds.

Error Messages

If some portion of the equipment fails to operate correctly, the computer will attempt to display an error message. The error messages are summarized in Table 3-1.

Table 3-1: Error Messages

```
+++ DISK ERROR: Drive not ready! +++
```

This error message usually is displayed when the system attempts to boot an operating system from the disk, but no disk has been inserted into the floppy disk drive, or the Winchester (if present and attempting to boot) has not been prepped (using the PREP utility) or formatted. This also may be caused by a faulty disk or data storage controller card. Be sure that there is a disk in the drive and that it is inserted correctly, with the drive latch closed properly.

```
+++ DISK ERROR: Bad disk controller! +++
+++ DISK ERROR: DMA Overrun! +++
```

These error messages usually indicate a fault in the data storage controller card, but may be caused by addressing clashes with other optional cards that are not supplied by Zenith Data Systems. It is also possible that the card is not seated properly.

```
+++ DISK ERROR: CRC error! +++
+++ DISK ERROR: Invalid address mark! +++
+++ DISK ERROR: Sector not found! +++
```

These error messages happen when booting an operating system from a disk. They can result from using an unformatted or defective disk (media), or from a faulty disk drive. Most often, this condition can be corrected by using another disk.

```
111 DISK ERROR: Seek failure! +++
```

This error message indicates that the computer cannot read the disk and/or cannot read track 0. This error can be caused by a faulty disk or by the failure of the data storage controller card, or the disk drive.

```
+++ DISK ERROR: Disk not bootable! +++
```

This error message occurs when the computer attempts to boot a disk that does not contain an operating system.

Table 3-1 (continued): Error Messages

```
+++ DISK ERROR: Invalid data read! +++
```

This error message indicates that the computer has read data from a disk but cannot use the data. In this case the data on the disk may have lost its integrity due to exposure to an external magnetic field, or due to physical damage of the disk itself. Try reading data from a known good disk and observe if the error is repeated.

```
+++ DISK ERROR: Data corrected! +++
```

This error message indicates that the drive has read data from a disk, but the data was not 100% defined as either logic 0 or logic 1. The data was adequately defined however, to allow the controller to determine the data bit's state.

```
+++ DISK ERROR: Cannot reset drive! +++
```

This error message indicates a probable fault in the disk drive. If possible, substitute the suspect drive with a known good one to isolate the problem.

```
+++ DISK ERROR: Must run SETUP to boot from Winchester! +++
```

This error message indicates that a valid drive type must be specified by the Setup program. Refer to the "Configuration" chapter in this part of the manual for the correct procedure to change the Setup program.

NOTE: The following error messages are related to Setup information that is stored in a special CMOS device. A battery backup circuit powers the device when the computer is turned off.

```
+++ ERROR: Please replace the backup battery! +++
```

This error message indicates that the real-time clock interface has lost power. This message will also appear upon the next powerup, after the I/O board has been removed from the computer backplane.

Table 3-1 (continued): Error Messages

```
+++ ERROR: Bad configuration information found in CMOS! +++
```

This message indicates that the CPU has found invalid data present in the special CMOS device. The Setup program will be automatically run by the computer after this error is detected. At this time, system configuration data may be determined and stored in the CMOS device.

```
+++ ERROR: Base memory size error! SETUP: 512K ACTUAL: 640K +++
```

The CPU always checks the size of memory indicated in the Setup program and compares it with the amount of actual memory installed in the computer. If the two figures are different, an error message similar to the one shown is generated. Be sure to update the Setup program if additional memory is installed in your computer.

```
+++ ERROR: Expansion memory size error! SETUP: 15296K ACTUAL: 0K +++
```

An error message similar to the one shown is generated if the CPU detects a difference between the amount of expansion memory indicated by the Setup program and the actual expansion memory installed. In the example, the actual memory is zero. This could mean that a major failure of memory devices has occurred or that the memory has been removed from the system.

NOTE: The following error messages are general messages that indicate particular errors in various areas of the computer.

```
+++ ERROR: Z-419 card not present! +++
```

This error message will occur if the user attempts to run the Z-248 computer in the Z-100 video graphics mode of operation, and the Z-419 card is not present in the computer system. Refer to Chapter 5, "Configuration" for additional information regarding the Z-100 mode of operation.

```
+++ ERROR: Memory parity failure! +++
+++ ERROR: System control processor failure! +++
```

These messages indicate that the system control processor on the I/O card is not responding to system requests. The keyboard and system will not function when this occurs.

Table 3-1 (continued): Error Messages

```
+++ ERROR: CMOS memory failure! +++
```

When this error message is generated it indicates that the CMOS RAM chip (real-time clock) cannot be properly read or written to.

```
+++ ERROR: CPU failure! +++
+++ ERROR: ROM checksum failure! +++
```

When either of these error messages occur, replace the CPU card.

```
+++ ERROR: Parity hardware failure! Address 00000:123D, Chip 209/U228! +++
```

An error message similar to the one shown indicates that the CPU is unable to read or write to the system RAM. The routine will attempt to display a number in the 200 range to indicate which chip has failed.

NOTE: Before replacing a card, check to see that the card is seated properly. If the computer has just been installed, check that it is properly configured for the amount of memory installed in the computer, as described in the "Configuration" chapter in this part of the manual.

```
+++ ERROR: Keyboard not responding or not connected! +++
```

Normally this error message is caused by the keyboard being unplugged. If this is not the case, the problem may be in the keyboard.

```
+++ ERROR: Timer interrupt failure! +++
```

This error message indicates possible failure of interrupt control and/or timer logic on the I/O card. Before replacing a card, make sure the card is properly seated and set up for the options installed. Also check that all optional cards are configured properly.

```
+++ Divide by zero! +++
```

This error message indicates an invalid parameter and returns program execution to the monitor program.

Table 3-1 (continued): Error Messages

```
+++ Non-maskable interrupt! +++
```

The NMI error message indicates a possible program execution error or presence of a power interruption.

```
+++ Overflow! +++
```

This error message occurs when the computer attempts to execute a mathematical calculation that exceeds the capabilities of the CPU.

```
+++ Wild interrupt! +++
```

This error message indicates that an interrupt generated by an unknown source was received. Program execution returns to the monitor program.

NOTE: The following information describes the meaning of the Error Summary Lines generated by the computer.

```
--- Errors found! Please press <ESC>  to continue ---
```

When an error has been detected in the system, this error summary message will be displayed on the last line of the screen. When the ESC (escape) key is pressed, the screen will clear and the diagnostics will continue.

```
--- Errors found! Please unlock keyboard, then press <ESC> to continue ---
```

This message will appear on the last line. The keyboard must be unlocked before the program will recognize that you have pressed the ESC key. When you press the ESC key, the screen will clear and the diagnostics will continue.

```
--- Fatal Error: Cannot Continue! ---
```

This message indicates that a major failure has occurred. The diagnostics will not continue under any circumstances when a fatal error is encountered. This error is most commonly caused by defective RAM at address 0000H.

Table 3-1 (continued): Error Messages

```
[Keyboard lock active: thekeyboard is disabled]
```

This message is displayed just before the monitor prompt, or just before the system boots the operating system from a disk. This is a warning to let you know that the keyboard is disabled by the keyboard lock. The keyboard will not work until it is unlocked.

Chapter 4

Disassembly

Z-248 Computer

The following procedures will help you gain access to particular hardware configuration jumpers and individual circuit cards. These procedures also can be used when installing optional circuit cards or assemblies in the computer.

NOTE: If one or more Winchester drives are installed in the computer, run the MS-DOS SHIP before you disassemble the computer.

Cover Removal

Refer to Figure 4-1 during the following procedure.

- Turn off the computer and unplug the line cord from the AC outlet.

WARNING: Hazardous voltages may be present inside the computer whenever the line cord is connected to an AC outlet. Do not begin disassembly before unplugging the unit.

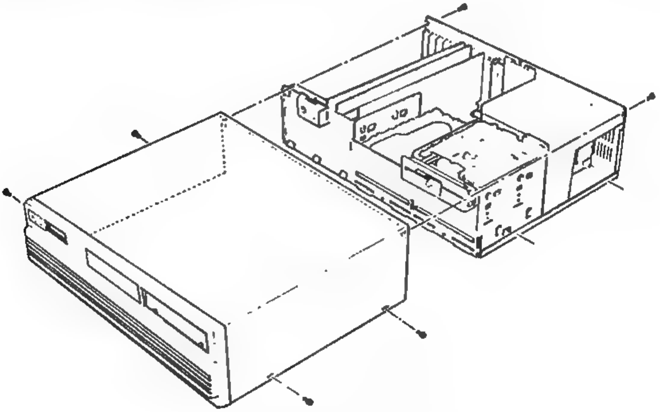


Figure 4-1: Cover Removal

- If a monitor is connected to the computer, disconnect its cable and set the monitor aside. Also disconnect the keyboard and any other peripherals connected to the computer.

- Make sure that the keyboard-disable lock is in the fully clockwise position (unlocked).

- Remove and save the six screws that secure the top cover to the unit.

- Remove the cover by sliding it to the front, while holding it up to avoid catching any wires, cables, or connectors, and set it to one side.

Card Removal and Replacement

CAUTION: Switch off the computer and unplug the power cord before you remove any cards. Cards can be damaged if the power is on when they are removed. To replace a card in the computer, follow this procedure (refer to Figure 4-2):

- Remove the computer cover.

- Remove the screw securing the card to be replaced (labeled A in Figure 4-2).

NOTE: Unplug any peripherals that are connected to the card through the back panel. Also disconnect any internal cables that are connected to the card, and any cables that are routed over the card.

- Be sure the card is disconnected from all internal and external cables, and then remove it from the backplane board by lifting straight up.

- To install a card, position the card edge connector directly over the slot on the backplane board and apply gentle downward pressure until the card is firmly seated. Reinstall the screw (labeled A in Figure 4-2) and connect the card to all appropriate internal and external cables.

- To add an optional card, remove the screw from the top of a blank bracket (labeled B in Figure 4-2). Do not discard the screw; you will need it to secure the optional card. Optional cards may be installed in any open slot in the backplane board.

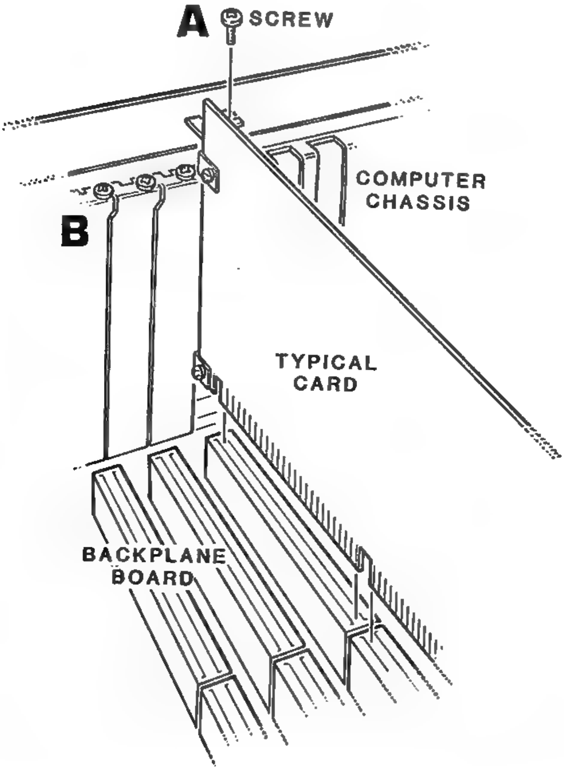


Figure 4-2: Card Support Brackets and Blank Brackets

Backplane Board Removal

CAUTION: Switch off the computer and unplug the line cord from the electrical socket before you remove the backplane board. It is necessary to remove all circuit cards from the backplane (refer to the preceding section). Unless power is removed, these cards could be damaged.

Refer to Figure 4-3.

- Remove all of the circuit cards as described in the Circuit Card Removal section of this chapter.

- Disconnect the keyboard cable from the DIN connector on the backplane board.

- Disconnect the power supply connector (P1) from the backplane board.

- Remove the battery from the backplane.

- Disconnect the backplane ground lead from the chassis of the computer with a short Phillips screwdriver. The ground lead is connected near the keyboard receptacle. Note that the power supply ground lead also is connected here.

- Remove the ten screws that secure the backplane board to the computer chassis.

- Lift the backplane board straight up and out of the computer.

- Be sure not to misplace the chassis spring retainer or chassis spring. The backplane board must be placed over these items when installed.

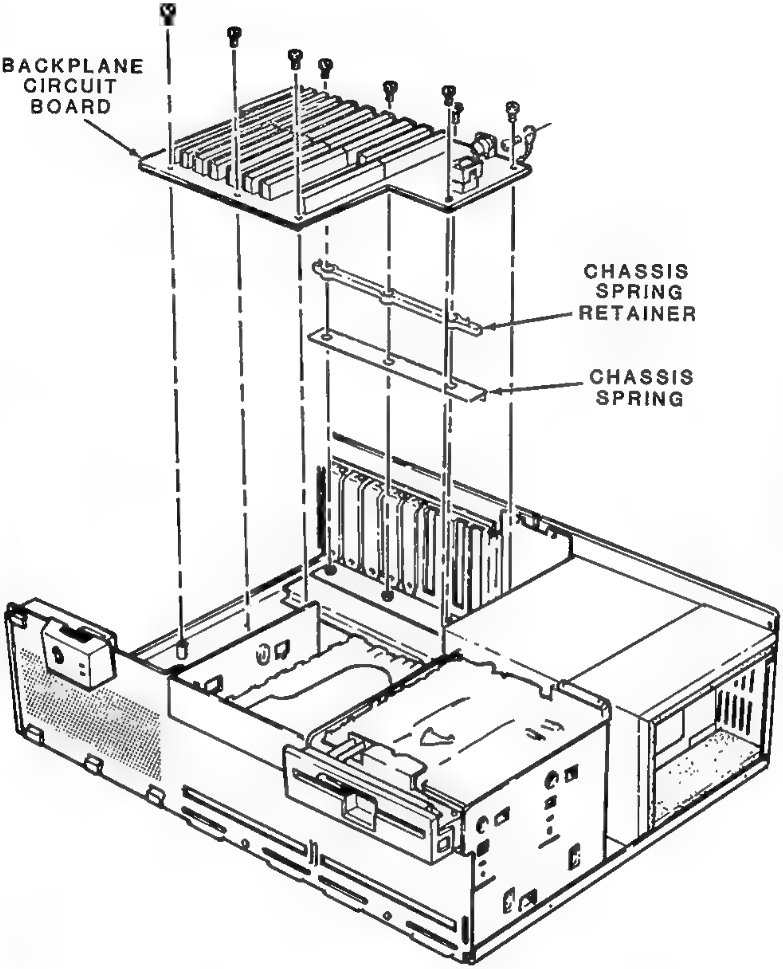


Figure 4-3: Backplane Board Removal

Power Supply Removal

Refer to Figure 4-4.

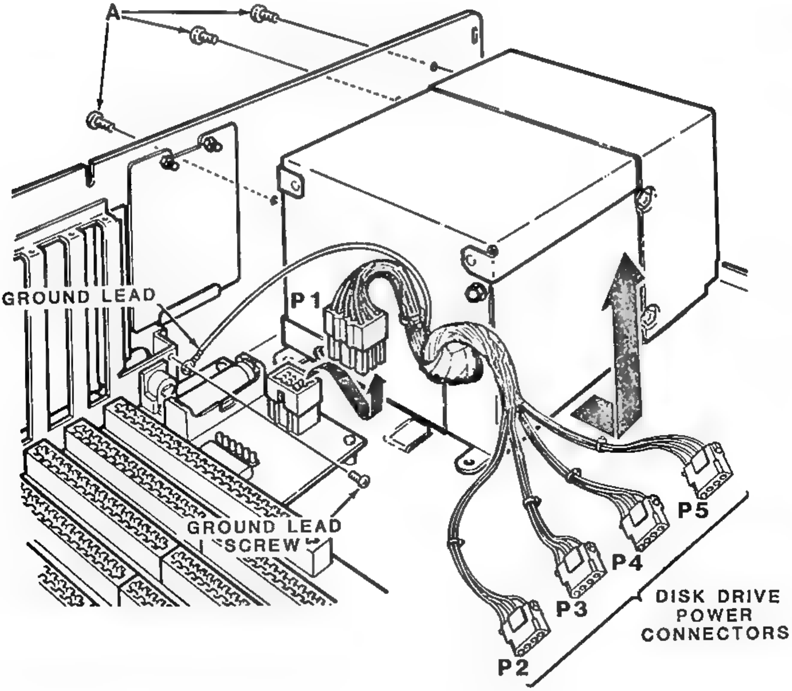


Figure 4-4: Power Supply Removal

- Switch off the computer and disconnect the line cord from the electrical socket.
- Disconnect the cables from the power supply at P1 (backplane board), and P2 through P5 (disk drives).
- Remove the battery from the backplane.
- Disconnect the power supply ground lead from the chassis of the computer with a short Phillips screwdriver. The ground lead is connected near the keyboard receptacle.

- Remove the three screws (labeled A) that secure the power supply to the back panel of the computer.
- Carefully slide the power supply one inch toward the front of the computer to release it from the bosses that retain it.
- Lift the power supply straight up and out of the computer.

Floppy Disk Drive Removal

CAUTION: If one or more Winchester drives are installed in the computer, run the MS-DOS SHIP utility before you remove the drive.

Drive A

Floppy disk drive A is located at the right front portion of the computer chassis. Optional Winchester drive 1 may be present beneath floppy drive A. To remove a drive, remove the bracket that contains and supports the drive as an assembly. Be sure to disconnect any applicable control and power cables from the drive. Remove the disk drive bracket assembly as follows:

- Refer to Figure 4-5 and remove the single screw (labeled A) that secures the drive bracket. Slide the drive bracket approximately one-half inch toward the rear of the computer, then lift the assembly carefully out of the computer chassis.

You can now remove four screws that secure floppy drive A to the drive bracket assembly, which will allow you to remove the drive. In addition, you can remove the four screws that secure optional Winchester drive 1 (if present) to the drive bracket, so you can take out the optional Winchester drive.

Drive B

Floppy drive B is located at the middle front portion of the computer chassis. Optional Winchester drive 0 may be present beneath floppy drive B. To remove a drive, remove the bracket that contains and supports the drive as an assembly. Be sure to disconnect any applicable control and power cables from the drive. Remove the disk drive bracket assembly as follows:

- Refer to Figure 4-5 and remove the single screw (labeled A) that secures the drive bracket. Slide the drive bracket approximately one-half inch toward the rear of the computer, then lift the assembly carefully out of the computer chassis.

You can now remove four screws that secure floppy drive B to the drive bracket assembly, which will allow you to remove the drive. In addition, you can remove the four screws that secure optional Winchester drive 0 (if present) to the drive bracket, so you can take out the optional Winchester drive.

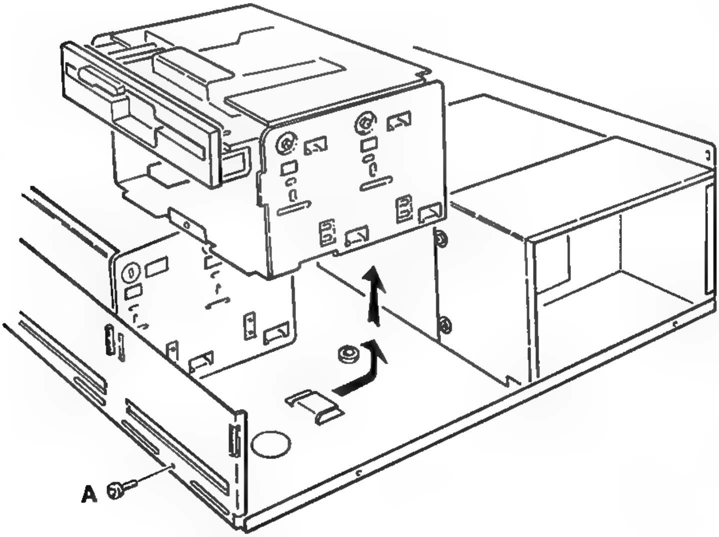


Figure 4-5: Typical Drive Bracket Assembly Removal

Winchester Disk Drive Removal

Drive 0

Optional Winchester drive 0 (if present), will be installed into the drive bracket assembly that also contains floppy drive B. Refer to the preceding "Floppy Disk Drive Removal" section explaining the removal of floppy drive B, for the procedure required to remove Winchester drive 0.

Drive 1

Optional Winchester drive 1 (if present), will be installed into the drive bracket assembly that also contains floppy drive A. Refer to the preceding "Floppy Disk Drive Removal" section explaining the removal of floppy drive A for the procedure required to remove Winchester drive 1.

Cover Replacement

Refer to Figure 4-6.

- Dress all of the cables down carefully so they are no higher than the PC cards or floppy drives.
- Make sure that the cables going to J5 on the floppy, Winchester, and tape backup controller card (slot 1), or to J5 on the floppy and tape backup controller card (slot 1), and to P301 on the I/O card (slot 4) are properly connected.
- Slide the cover back on the unit and secure it with the six screws removed and set aside during cover removal.
- Reattach the cables on the rear panel.

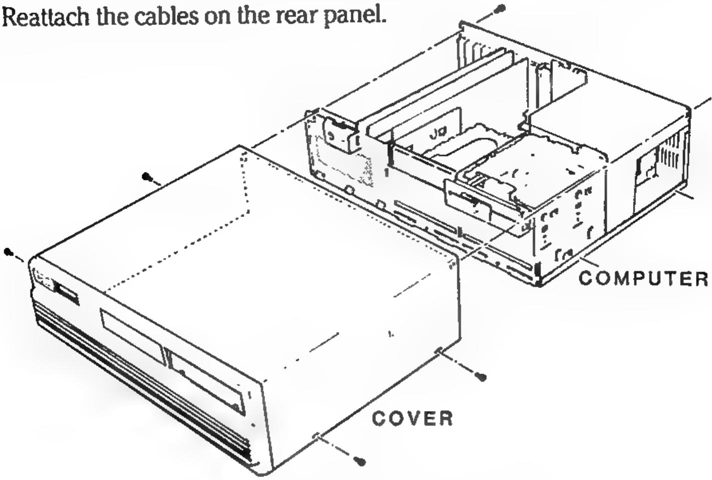


Figure 4-6: Cover Replacement

Tape Backup Unit

The tape backup unit is an external data storage device that is interfaced to the Z-248 computer through a multiple pin connector. Disassembly procedures for the tape backup unit will be found in that appendix in the back of this manual.

Chapter 5

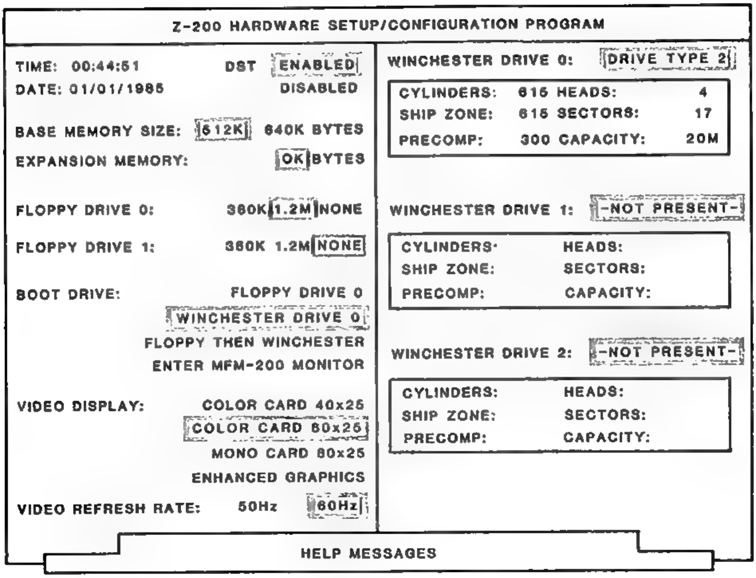
Configuration

This chapter defines the keyboard accessible setup program and hardware configuration jumpers used on each card. Information on standard configuration parameters (applicable to the Z-248) for floppy disk drives and Winchester disk drives also is provided.

Setup Program

The Setup program serves as a reference, giving the computer information about its system configuration. The information entered is kept in special CMOS memory for all future uses. The configuration data can easily be changed if you alter the computer's configuration. To run the program press and hold the **CTRL** and **ALT** keys, then press the **INS** key to drive the computer to the monitor program. Type **SETUP** after the monitor prompt (->) and press **RETURN**.

When the setup screen first appears, the standard selection in each field is highlighted with a rectangular, reverse-video image. Refer to Figure 5-1.



```
                 Z-200 HARDWARE SETUP/CONFIGURATION PROGRAM
TIME:  00:44:51          DST  ENABLED    WINCHESTER DRIVE 0:   DRIVE TYPE 2
DATE: 01/01/1985              DISABLED    CYLINDERS:  615 HEADS:        4
                                          SHIP ZONE:  615 SECTORS:     17
BASE MEMORY SIZE:  512K  640K BYTES       PRECOMP:    300 CAPACITY:   20M
EXPANSION MEMORY:             OK BYTES

FLOPPY DRIVE 0:         360K 1.2M NONE   WINCHESTER DRIVE 1:  -NOT PRESENT-
FLOPPY DRIVE 1:         360K 1.2M NONE    CYLINDERS·       HEADS:
                                          SHIP ZONE:       SECTORS:
BOOT DRIVE:              FLOPPY DRIVE 0   PRECOMP:         CAPACITY:
                     WINCHESTER DRIVE 0
                FLOPPY THEN WINCHESTER
                ENTER MFM-200 MONITOR    WINCHESTER DRIVE 2:  -NOT PRESENT-

VIDEO DISPLAY:     COLOR CARD 40x25       CYLINDERS:       HEADS:
                   COLOR CARD 80x25       SHIP ZONE:       SECTORS:
                    MONO CARD 80x25       PRECOMP:         CAPACITY:
                  ENHANCED GRAPHICS
VIDEO REFRESH RATE:    50Hz   60Hz
                             HELP MESSAGES
```

Figure 5-1: Setup Menu

You can move the highlighted image to the data field that matches your system's configuration using the cursor control keys. Use the space bar and BACK SPACE key to move between entries within each field, except the time and date fields.

To make a selection, move to the desired field and highlight the entry that describes your system. This selection will be locked in when you move to the next field. The message area at the bottom of the screen will provide help and error messages.

Time

The time and date are the only entries that need to be typed in. All the other data is selected using the highlighted cursor.

The time clock operates on a 24-hour basis. For example, 2 p.m. is 14:00 and 11 p.m. is 23:00. The format for entering the time is hh:mm:ss, where hh is the hour, mm is the number of minutes past the hour, and ss is the number of seconds into the minute.

The time must be entered in a 2-digit format. For example, the hour entry for 9 a.m. is 09. It is not necessary to enter the colons between the hours, minutes, and seconds. They will be entered automatically.

Example: 4:35 a.m. is entered as 04:35:00.
4:35 p.m. is entered as 16:35:00.

If the time is not entered correctly, an error message will appear at the bottom of the screen. The cursor will not move past the entry until a valid time is entered. If you make a mistake while typing, use the BACK SPACE key to return to the position where you made the mistake and type over it. After entering the time, press the **RETURN** key to move to the date entry.

Date

The date entry format is mm/dd/yyyy, where mm represents the month, dd represents the day of the month, and yyyy represents the year. Note that the numbers representing the month and day consist of two digits, while the year is represented by a 4-digit number.

Example: March 21, 1986 is entered as 03/21/1986.

It is not necessary to enter the slashes between the month, day, and year. They will be entered automatically. After entering the date, press the **RETURN** key and, using the right cursor control key, move to the DST field.

NOTE: The time and date fields are the only ones that are entered using the **RETURN** key. In the remaining fields, the selection will be locked in when you move from a field using the cursor control keys.

DST (Daylight Savings Time)

The next field, to the right of the date and time, is labeled DST, which stands for Daylight Savings Time. If you are operating the system in an area where Daylight Savings Time is in effect at some time during the year, you will want to enable this feature. The clock will automatically be set backwards or forwards one hour on the correct day.

Base Memory

Select the computer's base memory size by positioning the highlighted cursor over 512K or 640K, using the space bar and BACK SPACE key.

Expansion Memory

If the computer contains more than the maximum 640K of base memory, then you need to select the number of bytes of expansion memory you have added to the system. The selection ranges from 0 to 15,360 in increments of 64K. Use the space bar to make the number higher and the BACK SPACE key to choose a lower number. If you do not have expansion memory, set the number at 0.

Floppy Drive 0

If you are using one or more floppy disk drives, you need to choose between the 360K low-density drive, or the 1.2M high-density drive for each one that is present. Your first floppy drive is referred to as FLOPPY DRIVE 0. If you are not using a floppy drive, choose NOT PRESENT.

Floppy Drive 1

If you are using a second floppy disk drive, it is referred to as FLOPPY DRIVE 1. Choose again between the 360K drive and the 1.2M drive formats. If you are not using a second floppy drive, choose NOT PRESENT.

Boot Drive

The Boot Drive field allows you to set your default drive, which is the drive the computer will autoboot from. Four options are available:

- Option #1: The computer attempts to boot the first floppy disk.
- Option #2: The computer attempts to boot the first Winchester disk.

- Option #3: The computer attempts to boot the first floppy disk and if a bootable disk is not found, continues on to the first Winchester disk.

- Option #4: The computer enters the MFM-200 monitor program. The monitor prompt (->) will appear and you can then boot from any drive using the boot (B) command.

Video Display

Various video display adapters are available for use in Z-200 PC Series computers. These options are as follows:

- Color card: 40 × 25
- Color card: 80 × 25
- Monochrome card: 80 × 25
- Enhanced graphics

You must select the particular card that you intend to use as the default display adapter.

NOTE: The setup routine polls the actual video card that is installed in the computer. If problems are encountered after the default setting has been entered in the setup menu, it is possible that the polling routine could not accurately define the type of card present. If this occurs, the setup routine defaults to the J301 jumper setting on the I/O card to determine the type of video display adapter that is being used. J301 may be set to allow color graphics or monochrome display. Refer to the Hardware Jumper section of this chapter for information on jumper J301 (I/O card).

Video Refresh Rate

The video refresh rate should be set according to the power line frequency in your area. In the United States, the typical power line frequency is 60 Hz. If the system is not set correctly, the video screen will flicker.

Winchester Disk Drives

If one or more Winchester hard disk drives is installed in your computer, you need to select a drive "type" for each of them. You can determine the Winchester drive type by the code on the "DRIVE IDENTIFIER" label on the back panel of the computer. The drive identifier code is arranged in 3-digit sets with a maximum of 12 digits (4 drives). The 3-digit sets represent each drive as follows:

- The first 3-digit set represents the drive identifier code for floppy drive A.
- The second 3-digit set represents the drive type identifier code for either floppy drive B (if present) or for Winchester drive 0.
- The third 3-digit set represents the drive type identifier code for either Winchester drive 0 or Winchester drive 1, depending on whether floppy drive B is installed.
- The fourth 3-digit set represents the drive type identifier code for Winchester drive 1. The maximum of 12 digits indicates that four drives are installed in the computer.

Use Table 5-1 to reference the drive type identifier code to the specific drive type that you must enter in the SETUP menu.

Table 5-1: Z-200 Winchester Drive Types

DRIVE TYPE ID CODE	DRIVE TYPE	MANUFACTURER	MODEL	CAPACITY
156	1	Miniscribe	2012	10M
164	1	CMI	5412	10M
167	1	Seagate	412	10M
204	2	Seagate	225	20M
208	2	Seagate	4026	20M
237	2	CDC	9415-528	20M
215	7	Hitachi	DK511-5	40M
239	10	CDC	94155-48	40M

The Winchester drives indicated in Table 5-1 are currently approved for use in Z-200 PC Series computers. As additional drives are approved, applicable data will be supplied with the computer system.

If you are installing a supplementary drive and need to determine the drive type number, use Table 5-1 to cross-reference the model number of the drive to the correct drive type number. Some drives may have a label that indicates the drive type number. Also, applicable data may be included in the drive's package.

If running the PREP utility is required and you need information (number of cylinders, heads, precompensation cylinder, and shipping zone) about a specific drive in the computer, refer to Table 5-2.

Table 5-2: Winchester Disk Drive Data

TYPE	CAPACITY (BYTES)	CYL.	HEADS	PRECOMP	SHIPPING ZONE
1	10M	306	4	128	306
2	20M	615	4	300	615
3	30M	699	5	256	710
4	65M	940	8	512	940
5	49M	940	6	512	940
6	20M	615	4	FFFF	615
7	40M	699	7	256	710
8	57M	733	5	FFFF	733
9	117M	900	15	FFFF	901
10	40M	925	5	OOO	926
11	37M	855	5	FFFF	855
12	52M	855	7	FFFF	855
13	20M	306	8	128	319
14	44M	733	7	FFFF	733
15	0M	000	0	0000	000

NOTE: Winchester drives 1-14 each contain 17 sectors/track.

If the drive type is correctly specified in the SETUP menu, the PREP utility will automatically read the required information regarding the specific drive type.

NOTE: When you are through making changes or additions to the setup menu, press the **ESC** key and follow the prompts that appear in the help message area of the screen.

Slow Mode

The basic Z-248 computer runs with one wait state inserted into all memory access cycles. However, some applications programs that may be time-dependent may not execute correctly with only one wait state. To accommodate these applications programs, the Z-248 computer can be configured to operate with numerous wait states. If you have problems with some software packages, you may have to run the computer in the "slow" mode for the program to execute properly.

To configure the computer for the slow mode, enter o64,B1 (OUT command to port 64H) at the monitor prompt, then boot the operating system. The computer will automatically insert the required wait states into its memory access cycles. This time delay is approximately equal to two 8088 memory access cycles.

To return the computer to the fast mode of operation, enter o64,B2 at the monitor prompt, then boot the operating system.

Z-100 Mode

The Z-248 computer is capable of displaying Z-100 compatible graphics when the optional Z-419 bit-mapped video card is installed in the computer. This card provides an interface, via the extended graphics card, between the computer and color or monochrome monitors.

To configure the computer for the Z-100 video graphics mode of operation, boot the "Z-100 Mode DOS" version 3.1. This operating system will set all required parameters to accommodate the Z-100 mode as it boots. All DOS commands and utilities may be executed as for the standard DOS version 3.1.

To reconfigure the computer for standard Z-200 video output, boot the standard DOS version 3.1. The extended graphics card will automatically become the default video display adapter.

Hardware Jumpers

The following information describes the hardware configuration jumpers for the basic cards and disk drives used in the Z-200 PC Series Computers. Illustrations of each assembly are presented at the end of each section to help you locate the configuration jumpers.

NOTE: Configuration data for the (Zenith Data Systems) video interface and data storage controller card will be found in the appropriate appendix in the back of this manual. If a video interface other than the one indicated is intended to be used, refer to that cards specific documentation for configuration purposes.

If optional special performance cards are present in your computer, refer to each card's individual documentation for configuration purposes. If you require assistance in gaining access to the circuit cards, refer to Chapter 4, "Disassembly" in this part of the manual, that details the correct disassembly procedures.

CPU/RAM Card

CAUTION: This card contains electrostatic-sensitive devices (ESD). Handle these devices very carefully to prevent damage.

EPROM Device Selection (J201)

Jumper block J201 contains five separate pins used to configure EPROM addresses for particular EPROM devices. The default configuration has pins 1 and 2 shorted, and pins 3 and 4 shorted to allow 27256 (32 kilobit × 8) EPROM devices to be used. If pins 2 and 3 are shorted, and pins 4 and 5 are shorted, 27128 (16 kilobit × 8) EPROM devices may be used. The EPROMs are located at U214 – U215 on the CPU/RAM card.

Wait State Configuration (J202 – J206)

Jumpers J202 through J206 are used to configure the computer to operate with one wait state inserted into the memory access cycles (for PC AT compatibility) or without wait states. To operate the computer with zero wait states, faster DRAM devices must be used and delay line U250 must be changed. Various other discrete components will have to be changed to accommodate zero wait state operation. Configuration of these jumpers depends on system configuration.

Table 5-3 details those jumpers that must be configured for one wait state operation and those jumpers that must be configured for zero wait state operation. Figure 5-2 illustrates the location of the configuration jumpers on the CPU/RAM card.

Table 5-3: Wait State Configuration Jumpers

JUMPER	ONE WAIT STATE	ZERO WAIT STATES
J202	Removed.	Installed.
J203	Removed.	Pins 1 and 2 jumpered.
J204	Pins 1 and 2 jumpered.	Pins 2 and 3 jumpered.
J205	Pins 2 and 3 jumpered.	Pins 1 and 2 jumpered.
J206	Removed.	Installed

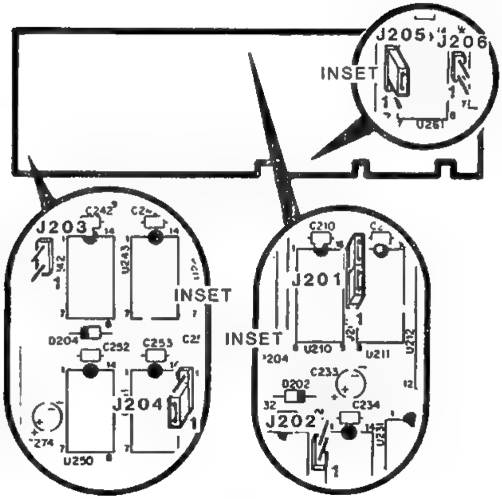


Figure 5-2: CPU/RAM Card Hardware Configuration

I/O Card

CAUTION: This card contains electrostatic-sensitive devices (ESD). Handle these devices very carefully to prevent damage.

CRT Adapter Select (J301)

This jumper is the CRT adapter select jumper. When this jumper is installed, it informs the system control processor that a color video circuit card will be used as the basic console video output. If jumper J301 is not installed, the system control processor provides for high-resolution monochrome as the basic console video output.

NOTE: The setup routine polls the actual display adapter that is installed in the computer. If for some reason the polling routine cannot define the type of display adapter present, the routine will default to the setting of jumper J301 to determine a default display. The default setting of J301 that should be used depends on the actual display adapter.

AC Voltage Present (J302)

When J302 is installed, it allows the status of the ACOK (AC voltage present) signal to trigger a non-maskable interrupt request to the CPU if voltage is lost or decreases sufficiently. If J302 is not installed, the ACOK signal is not capable of triggering a non-maskable interrupt. If AC voltage is lost, there will be a 5 ms delay before the potential (the DC voltage that powers the logic) diminishes. For the default setting, this jumper should be installed.

Parallel Port Interrupt Request (J303)

This jumper selects either IRQ5 (interrupt request 5) or IRQ7. If IRQ5 is desired, install a jumper across pins 2 and 3 of jumper block J303. If IRQ7 is desired, install a jumper across pins 1 and 2. The chosen signal is supplied to the programmable interrupt controllers.

Serial Port Interrupt Request (J304)

Jumper J304 selects either IRQ3 (interrupt request 3) or IRQ4 as the interrupt request initiated by the serial port. For IRQ3, short pins 2 and 3 with a jumper. If IRQ4 is desired, short pins 1 and 2 with a jumper. The resulting signal is supplied to the programmable interrupt controllers.

Parallel/Serial Port Address Select (J305)

J305 is the parallel/serial port address select jumper. This jumper is used to determine whether the parallel and serial ports will be addressed as LPT1 or LPT2 and COM1 or COM2 respectively.

Shorting pins 1 and 2 of J305 with a jumper provides port addresses for LPT1 and COM1. Installing a jumper across pins 2 and 3 of J305 provides port addresses applicable to LPT2 and COM2.

NOTE: All three jumpers (J303, J304, and J305) should be moved as a set for PC AT compatibility. Pins 1 and 2 of each jumper should all be shorted, or pins 2 and 3 of each jumper should all be shorted.

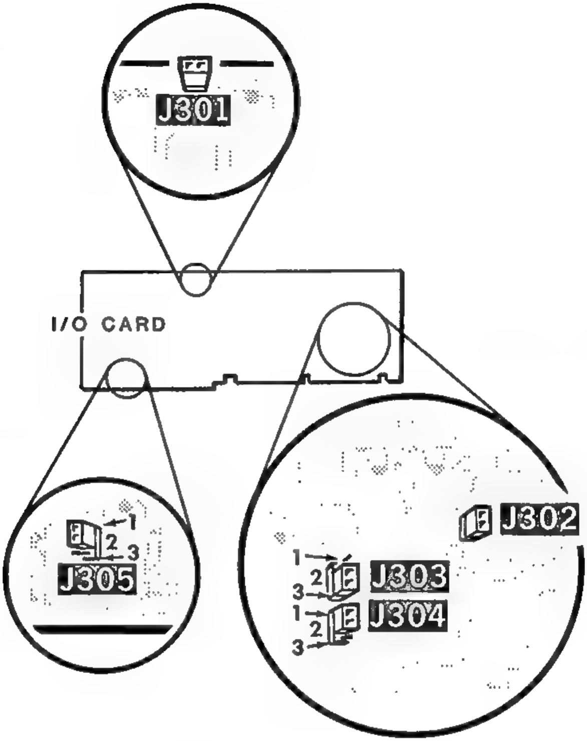


Figure 5-3: I/O Card Hardware Configuration

High Density (1.2M) Floppy Disk Drive

The following section details only those jumpers that should be installed on the 1.2M floppy disk drive. These jumpers represent the standard configuration of the drive when used in the Z-248 computer.

Drive Select Jumpers

The drive select lines enable one particular floppy drive, A or B, to be active at any given time. If two or more drives were enabled at the same time, loss of valuable data could occur. It is therefore necessary for the drive controller to arbitrate between drives to allow only one to be active at any time. Each drive is enabled with its own discrete binary address. Refer to Figure 5-4 for location of the standard settings (Z-248) of the drive select jumpers.

- DS1 — This drive select option must be configured with a jumper installed at DS1 for each floppy drive installed in the computer. If two drives are used they must be configured as DS1 because the interface cable has its drive select and motor control data lines (between the drives) twisted. The twist provides the correct logical address to each drive.

NOTE: The floppy drive connected to the last connector on the interface cable will always be drive A. In addition, a terminating resistor pack must be installed on the last drive attached to the interface cable (Figure 5-4). Viewed from the front of the computer, the floppy interface cable connectors should be attached to the floppy drives with the marked end of the cable facing to the right.

- TD — This option allows more than one drive to recognize the same drive select signal (DS1 for example). This condition causes the discrete terminating resistor connected to the drive select lines to be disconnected. However, a terminating resistor pack must be installed on the last drive connected to the floppy interface cable.

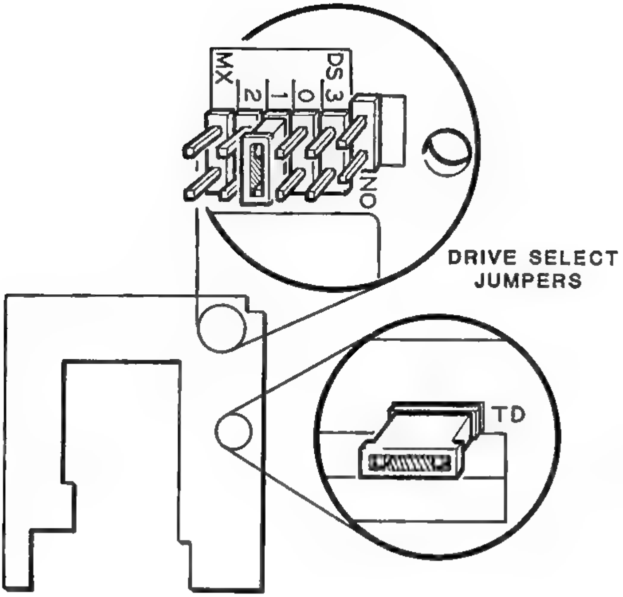


Figure 5-4: Drive Select Jumper Location

Motor Control Jumpers

Various ways of enabling the spindle motor are available as options. The drive controller design and timing requirements of the total system must be capable of supporting these options. Refer to Figure 5-5 for location of the standard settings (Z-248) of the motor control jumpers.

- MM — When the MM pins have a jumper installed on them (electrical short) and the MS pins do not (electrical open), the spindle motor will activate when a logical 0 is applied to the motor "on" input line. If a disk is not present in the drive, the motor will not activate under this configuration.
- SB — When the SS pins are open and the SB pins are shorted, the drive motor will always rotate at 360 rpm. During this mode of operation the "low write current" signal line must be brought low to allow correct recording onto a normal density floppy disk.

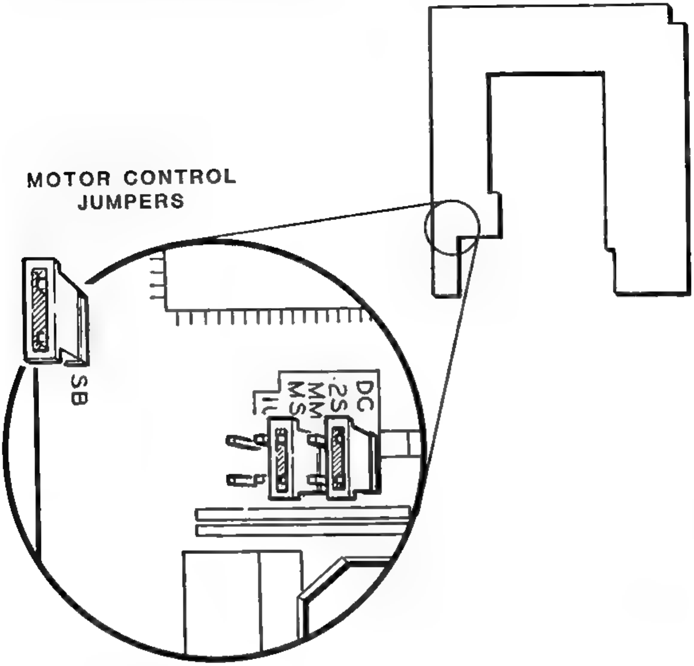


Figure 5-5: Motor Control Jumper Location

Head Load Jumpers

Many options are available to determine how the read/write heads load onto the disk. These options may be selected by jumpers. Refer to Figure 5-6 for location of the standard settings (Z-248) of the head load jumpers.

- HM — When the HM pins are shorted, the head will load when the motor "on" input goes low, a disk is in the drive, and the drive door is closed.

- UD — In order to eliminate unwanted oscillations of the head load solenoid, a delay must be introduced after the HM head load signal is made false. The head unload delay jumper (UD) enables this function when installed.

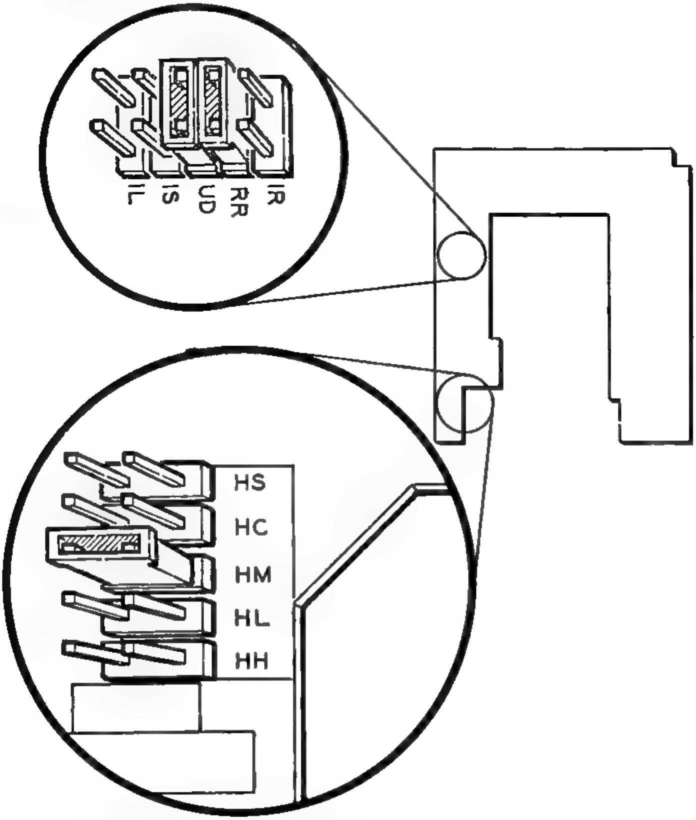


Figure 5-6: Head Load Jumper Location

Ready Status Jumpers

These jumpers enable the drive to output status signals to the data storage controller card. The status signals inform the controller when a disk is in the drive, when a disk change operation is occurring, and when a disk is not in the drive. Refer to Figure 5-7 for location of the standard settings (Z-248) of the ready status jumpers.

- DC — When the DC pins are shorted by a jumper and the 2S pins are left open, the drive ready output will be a logical 1 whenever a disk is properly installed in the drive. The drive status is presented on the high-to-low transition of the drive select signal.
- RR — This radial output ready setting causes the ready output of the drive to always be enabled. The drive select signal will not have an effect on the ready output status when this jumper (RR) is installed.

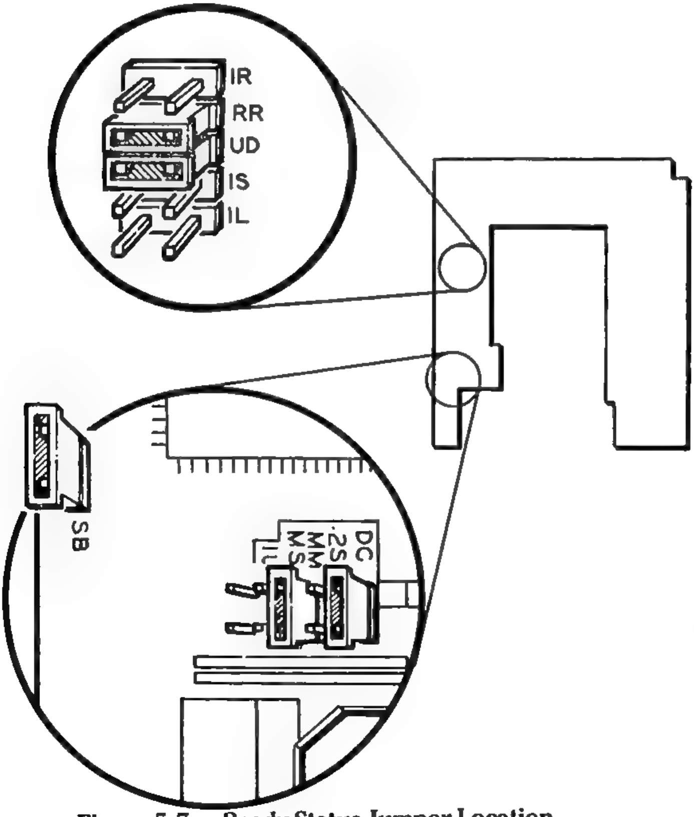


Figure 5-7: Ready Status Jumper Location

LED Indicator Status Jumper

The LED that is mounted in the front bezel of the high density drive provides a visual indication when that drive is active. The LED will light during read and write operations. Refer to Figure 5-8 for location of the standard setting (Z-248) of the LED indicator status jumper.

- IH1 — When the pins at this jumper location are shorted, the LED in the front bezel of the drive will light whenever the read/write heads in that drive are loaded onto the disk.

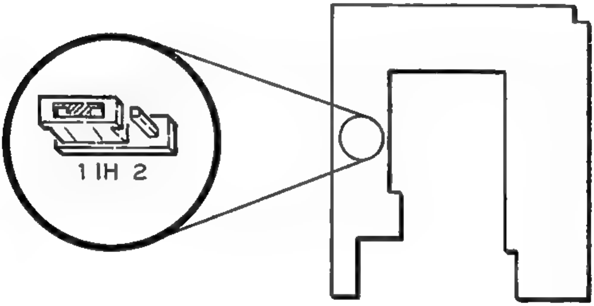


Figure 5-8: LED Indicator Status Jumper Location

Standard Density (360K) Floppy Disk Drive

Various types of standard density (360K) floppy drives are available for use in the Z-200 PC Series Computer. Each floppy drive must have its drive select inputs configured as DS1 because the interface cable has its drive select and motor control data lines (between the drives) twisted. The twist provides the correct logical address to each drive.

NOTE: The floppy drive connected to the last connector on the interface cable will always be drive A. In addition, a terminating resistor pack must be installed on the last drive attached to the interface cable (Figure 5-9). Viewed from the front of the computer, the floppy interface cable connectors should be attached to the floppy drives with the marked end of the cable facing to the right.

Refer to Figure 5-9 during the following discussion.

There are two types of 360K floppy drives that require modification so they operate correctly in the Z-248 computer. Other types of drives may require different modification or configuration. Refer to the appropriate documentation supplied with those drives for specific information.

For drive type 1, the trace leading to pin 34 must be opened. Carefully cut the trace using a small sharp knife. For drive type 2, the trace labeled RY must be opened. Carefully cut the trace using a small sharp knife.

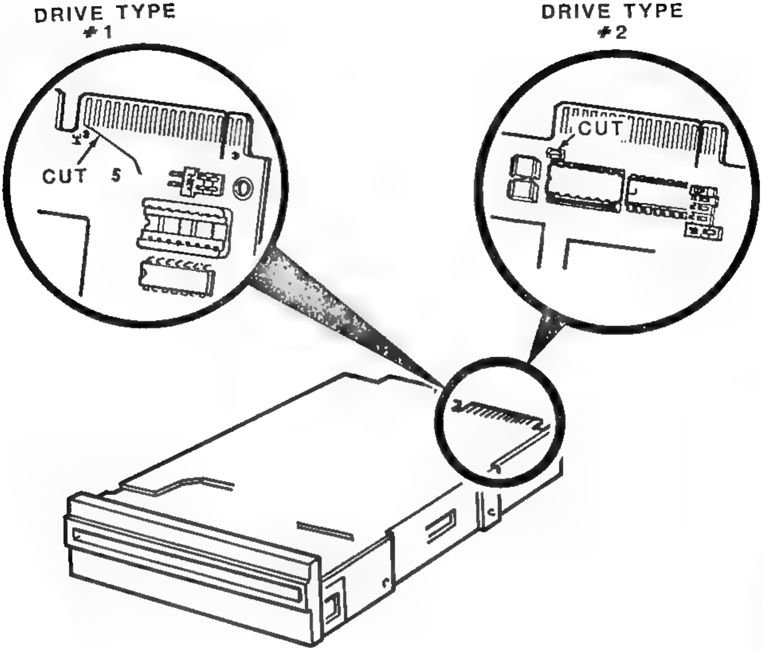


Figure 5-9: 360K Floppy Hardware Configuration

Winchester Disk Drives (20M and 40M)

These Winchester disk drives (if used) must have their drive select jumpers configured as DS1. In all cases, the drive referred to as drive 0 must be attached to the last connector on the interface cable. In addition, a terminating resistor pack must be installed on this drive.

Both drives (when used) are configured as DS1 because the interface cable has its drive select and motor control data lines (between the drives) twisted. This provides the correct logical address to each drive.

NOTE: The Winchester drive connected to the last connector on the interface cable will always be drive 0. In addition, a terminating resistor pack must be installed on the last drive attached to the interface cable (Figure 5-10). Viewed from the front of the computer, the 34-pin and 20-pin Winchester interface cable connectors should be attached to the Winchester drives with the marked end of the cable facing to the left.

Refer to Figure 5-10 for location of the drive select jumpers and terminating resistor pack for the 20M and 40M Winchester disk drives.

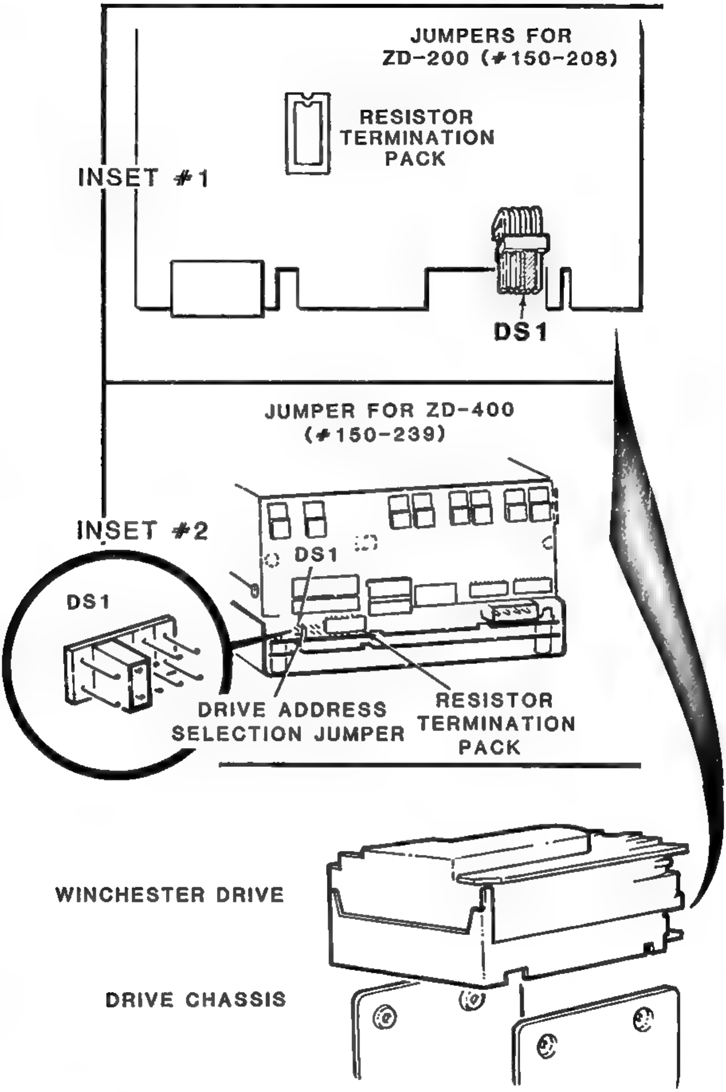


Figure 5-10: Winchester Configuration Jumpers (20M and 40M)

Part III

System Programming

The Z-200 Series Computers use firmware-established interrupts for most control and input/output functions. To maintain a high level of software compatibility with PC-compatible computers, you will find the interrupts in the Z-200 PC Series to be largely identical to those found in the Z-100 PC Series.

This part of your Technical Manual provides information concerning the use of the interrupts established and managed by the MFM-200 Monitor Program.

Chapter 6
Software Interface

MFM-200 Functions

During the initial power-up sequence, the MFM-200 performs a number of functions, including a self-test of all circuits in the system to make sure the computer is ready to function. If a malfunction is detected, one or more messages from a comprehensive set is sent to the system's output device (usually a video card) to alert the operator. For a complete discussion of the self-test function and possible error messages, see Chapter 3, "Firmware."

The SETUP command, part of the MFM-200, allows the user to specify the configuration of the computer, including the current time and date for the clock, the amount of installed memory, the number and type of floppy disk drives, the number and type of Winchester disk drives, the type of video display card installed, and the power line frequency. Chapter 5 contains a complete discussion of the SETUP function of the MFM-200.

The other functions of the MFM-200 include system initialization, screen input/output, disk input/output, keyboard interface, parallel and serial input/output, diagnostics, and system control routines. These are modularized for easier maintenance and improved reliability.

Finally, there are two jump vectors in the MFM-200 ROM, which can be used when creating applications programs. They are located at F000:FFF0, which is the power-up reset vector, and F000:FFED, which is an unconditional jump to the MFM-200 monitor program (the monitor program prompt will appear on the screen).

MFM-200 Operation

The firmware contains the machine-level code which provides control, at the device level, of the main input/output devices of the computer. To perform a function, a specific code is placed in register AH of the CPU and the INT instruction is executed. As a rule, these MFM-200 routines will preserve the values in all CPU registers except for AX and the flags. Other registers may be modified upon completion of the task only if they are returning a value to the program that executed the function.

The firmware also generates software interrupts when certain events occur. To gain access to these events, you will have to patch the interrupt vectors involved with the events for your routines. If you do this, you may experience unexpected results when you run your routines with other programs that use or modify the same interrupts or memory locations. Table 6-1 provides a map of the system's addressable memory. Table 6-2 is a map of the system ports.

NOTE: The information presented in these tables is subject to change as new products are introduced for this family of computers.

Table 6-1: System Memory Use

ADDRESS RANGE	SIZE	BUS WIDTH	DESCRIPTION OF USE
000000H – 07FFFFH	512K	16 Bits	Base memory
080000H – 09FFFFH	128K	16 Bits	Expansion memory
0A0000H – 0BFFFFH	128K	8 Bits	PC-compatible video
0C0000H – 0DFFFFH	128K	8 Bits	Expansion ROM
0E0000H – 0EFFFFH	64K	16 Bits	Reserved for system ROM
0F0000H – 0F0FFFH	4K	8 Bits	Scratchpad video RAM
0F1000H – 0FFFFFH	60K	16 Bits	System ROM
100000H – FDFFFFH	15,232K	16 Bits	Protected memory
FE0000H – FEFFFFH	64K	16 Bits	Mapped from 0E0000 – 0EFFFF — Reserved for system ROM
FF0000H – FF0FFFH	4K	8 Bits	Mapped from 0F0000 – 0F0FFF — Scratchpad video RAM
FF1000H – FFFFFFH	60K	16 Bits	Mapped from 0F1000 – 0FFFFF — System ROM

Table 6-2: System Port Map

ADDRESSES	DESCRIPTION OF PORT
000H – 00FH	DMA processor
020H	Interrupt mask register
021H	Interrupt command register
040H – 043H	System timer
060H – 063H	PPI status port
080H – 083H	DMA page registers
0A0H – 0AFH	Non-maskable interrupt registers
0C0H – 0CFH	Reserved
0E0H – 0EFH	Reserved
200H – 20FH	Game input/output port
278H – 27FH	Reserved
2F8H – 2FFH	Reserved
378H – 37FH	Parallel printer #1
3B0H – 3BFH	Monochrome monitor
3D0H – 3DFH	Color graphics controller
3F0H – 3F7H	Floppy disk controller
3F8H – 3FFH	Serial input/output interface

Using an Interrupt

Any interrupt that has not been initialized will jump to a return (RET) function (no operation). Some of the interrupts discussed in this manual are initialized by the operating system. Since MS-DOS is a widely-used operating system for these computers and helps maintain PC-compatibility, those interrupts that are considered universal (that is, not specific to a particular version of MS-DOS) are included in this manual and noted in Table 6-3. Those based on MS-DOS that are subject to change between versions are discussed in the Programmer's Utility Pack for MS-DOS. Other operating systems may or may not initialize similar interrupts.

Some of the interrupt routines are not recommended for general use. The reason these instructions are retained in the Z-200 PC Series computers rather than deleted is that they serve to maintain PC AT software compatibility. Generally, the interrupt instructions that you will find more useful are referenced in the appropriate text.

When you prepare to use or modify an interrupt routine, there are a couple of factors you should keep in mind.

- All parameters that are passed to and from MFM-200 must go through the registers in the 80286 CPU. Where more than one function is possible, or where additional parameters are required, more than one register is used.

- Most of the interrupt routines preserve the values in the registers except when a value is being returned to a specific register. When you write routines for these interrupts, you should plan on preserving, where possible, the values of the CPU registers.

Each interrupt routine is pointed to by a dedicated interrupt vector. To execute one of the interrupt routines, you must first load the specified parameters into the registers of the CPU. Then perform an INT xxH instruction where xxH is the interrupt number. For instance, the example in Listing 6-1 executes the INT 11H interrupt and then tests for and returns the number of drives connected to the system as established by the SETUP routine.

Listing 6-1: Sample Code

```
                                  ;test for number of drives
     INT    11H                   ;retrieve status byte
     PUSH   AL                    ;temporarily store results
     AND    AL,1B                 ;isolate bit 1
     TEST   AL,0                  ;test for drives present
     JZ     ZERO_DRIVES           ;if no drives, jump to routine
     POP    AL                    ;pop results back off stack
     AND    AL,1100000B           ;isolate bits 5 & 6
     MOV    CL,5                  ;set for shift of 5 bit-places
     SHR    AL,CL                 ;shift data right 5 bits
     INC    AL                    ;increment results by 1 so
                                  ;that true number of drives
                                  ;is now in register AL
```

Since no parameters were required by INT 11H, none were loaded. Note however, that the results were returned in register AL, where they were tested by the routine.

Modifying an Interrupt

To modify an interrupt, you must patch the interrupt vector with the address of your routine as outlined in the following procedure (MS-DOS operating system environment).

1. Use the MS-DOS Function Request Get Interrupt Vector (35H) to read the current value of the interrupt vector. Save this value in your code segment.

2. Use the MS-DOS Function Request Set Interrupt Vector (25H) to set the address of your interrupt routine.

3. Do **not** execute an IRET instruction at the end of your routine. Instead, execute a JMP DWORD PTR to the address saved in step 1. This allows multiple background tasks to have a chance at the interrupt. Note that the final routine executed during an interrupt sequence is an IRET, but only after the CPU registers have been cleaned up.

4. Do **not** modify any CPU registers unless specifically stated in the discussion of that interrupt. Upon entry to an interrupt routine, the ES, DS, BX, CX, DX, AX, BP, SI, and DI registers contain indeterminate information that must be preserved. The information will be in a stack preserved by the BIOS, and is sufficient for most purposes. However, it is a good practice to set up your own local stack, which can then be restored just prior to exiting your routine.

Software Interrupt Summary

Table 6-3 summarizes the interrupts and describes the function, what initializes it, and the chapter where you can find a complete discussion. Interrupts followed by dashes (—) in each column are either reserved for future use or are not generally used by PC-compatible computers and more specifically, they are not implemented in Zenith Data Systems computers or software.

Table 6-3: Interrupt Summary

INTERRUPT NUMBER	INITIALIZED BY	DISCUSSION CHAPTER	FUNCTION NAME
00H	DOS	7	Divide by Zero
01H	DOS	7	Single Step
02H	MFM-200	7	Non-Maskable Interrupt
03H	DOS	7	Software Breakpoint
04H	DOS	7	Arithmetic Overflow
05H	MFM-200	9	Print Screen
06H	—	—	—
07H	—	—	—
08H	MFM-200	7	Timer (Time-of-Day)
09H	MFM-200	8	Key Pressed
0AH	MFM-200	7	Second 8259 Programmable Interrupt Controller
0BH	Software	9	Communications (COM2)
0CH	Software	9	Communications (COM1)
0DH	Software	9	Alternate Parallel Printer (LPT2)
0EH	MFM-200	10	Floppy Disk Drive
0FH	Software	9	Parallel Printer (LPT1)
10H	MFM-200	11	Video Input/Output
11H	MFM-200	7	Equipment Configuration
12H	MFM-200	7	Memory Size
13H	MFM-200	10	Disk Input/Output
14H	MFM-200	9	Serial Input/Output
15H	MFM-200	7	Device Control
16H	MFM-200	8	Key Pressed
17H	MFM-200	9	Printer Input/Output
18H	MFM-200	9	Parallel/Serial Configuration
19H	MFM-200	10	Booting an Operating System
1AH	MFM-200	7	Set/Read Time-of-Day
1BH	MFM-200	8	Keyboard Input/Output
1CH	MFM-200	7	Timer Tick
1DH	MFM-200	11	Video Initialization
1EH	MFM-200	10	Disk Parameters
1FH	MFM-200	11	Defining Characters
70H	MFM-200	7	Real-Time Clock
71H	MFM-200	7	Software redirected to IRQ2
72H	—	—	—
73H	—	—	—
74H	—	—	—
75H	MFM-200	7	80287 Non-Maskable Interrupt
76H	Winch. ROM	10	Rigid Drive Controller
77H	—	—	—

For convenience, Table 6-4 (CPU Interrupts), Table 6-5 (Keyboard Interrupts), Table 6-6 (I/O Interrupts), Table 6-7 (Data Storage Interrupts), and Table 6-8 (Video Interrupts) group the interrupts that pertain to specific areas of the computer together. Each table is repeated in the chapter that discusses those interrupts.

Table 6-4: CPU Interrupts

INTERRUPT	FUNCTION
00H	Divide by Zero
01H	Single Step
02H	Non-Maskable Interrupt (NMI)
03H	Software Breakpoint
04H	Arithmetic Overflow
08H	Timer (Time-of-Day)
0AH	Second 8259 Programmable Interrupt Controller
11H	Equipment Configuration
12H	Memory Size
15H	Device Control
1AH	Set/Read Time-of-Day
1CH	Timer Tick
70H	Real-Time Clock
71H	Software redirected to IRQ2
75H	80287 Non-Maskable Interrupt

Table 6-5: Keyboard Interrupts

INTERRUPT	FUNCTION
09H	Key Pressed
16H	Keyboard Input/Output
1BH	Keyboard Break

Table 6-6: I/O Interrupts

INTERRUPT	FUNCTION
05H	Print Screen
0BH	Communications (COM2)
0CH	Communications (COM1)
0DH	Alternate Parallel Printer (LPT2)
0FH	Parallel Printer (LPT1)
14H	Serial Input/Output
17H	Printer Input/Output
18H	Parallel/Serial Configuration

Table 6-7: Data Storage Interrupts

INTERRUPT	FUNCTION
0EH	Floppy Disk Drive
13H	Disk Input/Output
19H	Booting an Operating System
1EH	Disk Parameters
76H	Rigid Drive Controller

Table 6-8: Video Interrupts

INTERRUPT	FUNCTION
10H	Video Input/Output
1DH	Video Initialization
1FH	Defining Characters

Hardware Interrupts

Although this part of the technical manual is concerned about the software interrupts, a distinction needs to be made between them and interrupt requests made by the hardware.

The two 8259 programmable interrupt chips are used to manage hardware interrupts. Because more than one request for an interrupt can occur at the same time, the two chips work together to manage these requests on a priority basis.

The highest priority is assigned to IRQ0 (interrupt request 0) and the lowest to IRQ7. IRQ8 – IRQ15 are assigned priorities following IRQ2. By sending a mask through port 021H, the programmer can control whether a particular interrupt can be processed when requested. For more information on the 8259 programmable interrupt controller, refer to Chapter 14, “Input/Output Card in Part IV of this manual. Table 6-9 contains a description of the usual hardware requests and the software interrupt they trigger, in the order of precedence. Jumpers and/or other hardware factors may change the source of the interrupt request (not recommended).

Table 6-9: Hardware Generated Interrupt Requests

HARDWARE IRQ	SOFTWARE INT	USUAL SOURCE OF INTERRUPT
0	08H	Timer (Time-of-Day)
1	09H	Key pressed
2	0AH	Second 8259 programmable interrupt controller
8	70H	Real-time clock
9	71H	Software redirected to IRQ2
10	72H	Reserved
11	73H	Reserved
12	74H	Reserved
13	75H	NMI Numeric processor extension (80287)
14	76H	Rigid drive controller
15	77H	Reserved
3	0BH	Communications (COM2)
4	0CH	Communications (COM1)
5	0DH	Alternate parallel printer (LPT2)
6	0EH	Floppy disk controller
7	0FH	Parallel printer (LPT1)

Chapter 7

CPU Interrupts

Programming CPU Interrupts

Those interrupts that are generally considered to be CPU interrupts are defined in Table 7-1. For information pertaining to the use and programming of interrupts, see Chapter 6, "Software Interface."

Table 7-1: CPU Interrupts

INTERRUPT	FUNCTION
00H	Divide by Zero
01H	Single Step
02H	Non-Maskable Interrupt (NMI)
03H	Software Breakpoint
04H	Arithmetic Overflow
08H	Time-of-Day Timer
0AH	Second 8259 Programmable Interrupt Controller
11H	Equipment Configuration
12H	Memory Size
15H	Device Control
1AH	Set/Read Time-of-Day
1CH	Timer Tick
70H	Real-Time Clock
71H	Software redirected to IRQ2
75H	80287 Non-Maskable Interrupt

Divide by Zero (INT 00H)

INT 00H (divide by zero) will be executed if a divide instruction produces a quotient that is too large to fit in the result register (such as dividing a value by 0). The error routine will print `Divide Overflow` and return control to the operating system. This routine is initialized by the DOS. If you intend to use this routine other than with the operating system, you must set up the vector to intercept DIV and IDIV instructions that can result in exceptions to the rule (for instance, BASIC does this to retain control inside BASIC).

Single Step (INT 01H)

INT 01H (single step) is used for executing a single machine instruction at a time. It is called by the 80286 when an instruction is executed with the trace flag (TF) set. It is most commonly used by routines such as the MS-DOS DEBUG command and the MFM-200 trace command. Since this routine is initialized by the command (MS-DOS or MFM-200) calling it, you must define the routines to be executed when you establish the conditions and routines that will call this interrupt.

Non-Maskable Interrupt (INT 02H)

INT 02H is the non-maskable interrupt (NMI). It is initiated by hardware external to the 80286, but in these computers, it usually indicates a power-down condition has started. Although it is off at power-up, it usually is not disabled except by design or by sending 00H to the input/output port A0H. To enable the NMI, send 80H to the input/output port A0H.

NOTE: The 80287 Numeric Processor Extension uses this interrupt in its normal operation.

Software Breakpoint (INT 03H)

You may run a program until the processor encounters a breakpoint (an INT 03H instruction). Usually, the MFM-200 debugging routines or MS-DOS DEBUG will allow the user to set up breakpoints and subsequently will handle the condition accordingly (by usually returning control to the last command level).

Arithmetic Overflow (INT 04H)

The INT 04H instruction is executed from an INTO instruction when the overflow flag (OF) is set. The flag is set (usually) by a previous arithmetic or logic operation.

Time-of-Day Timer (INT 08H)

The INT 08H instruction is used to keep track of the time-of-day for the system. It also calls the INT 1CH instruction (Timer Tick).

The 8253 Programmable Interval Timer initiates this interrupt 18.2159 times per second, or every .054897095 seconds. This timer is not affected by the CPU clock speed and is used for such functions as keeping track of the time-of-day, timing out the disk motors, and calling the timer tick interrupt (1CH).

The timer keeps track of the time-of-day in a 32-bit word (sometimes called a double-word). When the count reaches approximately 1803D8H (1,573,848 decimal), a flag is set to 1 to indicate that the timer has rolled past midnight to a new day. Interrupt 1AH is used to set and/or read the value of this word. This interrupt is established by the hardware and is IRQ0.

Second 8259 Interrupt Controller (INT 0AH)

The 8259 programmable interrupt controllers manage hardware interrupt requests generated by the hardware in the computer. Refer to Chapter 6 for a table of the interrupt request (IRQ) numbers and their equivalent software interrupts. Interrupt 0AH is used internally by MFM-200 to control the second 8259 programmable interrupt controller. Do not use this interrupt in your programs.

Equipment Configuration (INT 11H)

The INT 11H instruction can be used to report back the configuration of the equipment as established by SETUP and stored in CMOS RAM (see Chapter 5, “Configuration”) and any jumpers on CPU/RAM, I/O, video, and/or data storage controller cards. The report is returned in a 16-bit word (register AX). Since this interrupt is common to PC AT compatible equipment, you need to be aware of all possible responses and what they mean. Refer to Table 7-2 for the definitions of each bit of the word returned in register AX.

Table 7-2: Equipment Configuration (Register AX Report from INT 11H)

BIT NO.	DESCRIPTION
0	Floppy disk drives are installed in the system if set to 1.
1	The 80287 numeric processor extension is installed in the system if set to 1.
2 and 3	Amount of base RAM (device size) installed in memory.
0 0	Unused in Zenith Data Systems computers. 16-kilobit devices in some computers.
0 1	Unused in Zenith Data Systems computers. 32-kilobit devices in some computers.
1 0	Unused in Zenith Data Systems computers. Rarely used in any system. 48-kilobit devices in some computers.
1 1	64- or 256-kilobit devices in Zenith Data Systems computers (not specific).
4 and 5	Initial video mode at power-up.
0 0	Enhanced graphics card installed as primary display.
0 1	Unused in Zenith Data Systems computers.
1 0	80 × 25 monochrome on the color board (RGB output).
1 1	80 × 25 monochrome on the Z-329 monochrome board (monochrome output).
6 and 7	Reports the number of floppy disk drives installed in the computer if bit 0 is set.
0 0	1 drive.
0 1	2 drives.
1 0	Unused in the Z-200 PC Series Computers. Reports 3 drives in other computers.
1 1	Unused in the Z-200 PC Series Computers. Reports 4 drives in other computers.
8	Unused in Zenith Data Systems computers.
9, 10, and 11	Reports the number of RS-232 ports in the system in binary format: The standard input/output on this system emulates the PC AT input/output, so the minimum will be one port (0 0 1).

Table 7-2 (continued): Equipment Configuration (Register AX Report from INT 11H)

BIT NO.	DESCRIPTION
12	If a game input/output card (adapter) is installed, for using joysticks, paddle controllers, etc., this bit will be set to 1.
13	Unused by Zenith Data Systems computers.
14 and 15	Reports the number of printers installed in binary format.

Memory Size (INT 12H)

The INT 12H instruction returns in register AX the number of contiguous 1K blocks of user memory in the system, as established by data found in CMOS RAM. For instance if the value in AX is 256 following this interrupt, there is 256K of memory installed in the system.

Device Control Interrupt (INT 15H)

The device control interrupt (15H) allows the programmer various functions such as placing the 80286 CPU into protected mode, accessing memory above the 1 megabyte real mode limit, using the joysticks, and so on. The particular function chosen depends on the value of the AH register when the interrupt is executed. Table 7-3 lists the functions. Those that are marked as "user trappable" indicate that application programs do not normally make these function requests and, therefore, user programs may intercept interrupt 15H and trap these functions as needed. Regardless of the use of these function made by other software, MFM-200 will execute these functions at the times described in the text following the table.

Note that "devices" and "processes" are defined by softrware, most notably an operating system environment, such as that provided by MS-DOS or XENIX.

Table 7-3: Interrupt 15H Functions

FUNCTION (AH) CODE	DESCRIPTION
00H – 7FH	Reserved — unused by MFM-200.
80H	Device open — user trappable.
81H	Device close — user trappable.
82H	Program terminate — user trappable.
83H	Event wait.
84H	Joysticks support.
85H	System request key — user trappable.
86H	Wait.
87H	Block move.
88H	Determine extended memory size.
89H	Place processor in virtual mode.
8AH – 8FH	Reserved — unused by MFM-200.
90H	Device busy — user trappable.
91H	Set interrupt complete flag — user trappable.
92H – FFH	Reserved — unused by MFM-200.

Function Code 80H: Device Open — MFM-200 will generate this interrupt whenever a device is opened. The device ID (identification) is placed in register BX and the process ID is placed in register CX. Both identification codes must be defined in the operating system environment or the program. Normally, this handler provides only an RET operation. However, this interrupt can be used to detect open devices.

Function Code 81H: Device Close — MFM-200 will generate this interrupt whenever a device is closed. The device ID is placed in register BX and the process ID is placed in register CX. Both identification codes must be defined in the operating system environment or the program. Normally, this handler provides only an RET operation. However, this interrupt can be used to provide the process ID. This interrupt can be used to detect closed devices.

NOTE: Both function code 80H and 81H require that the same information be placed in register BX and register CX. The most common use for these two functions would be in a multiuser or multifunction environment where detection of open and closed devices is critical to dependable operation of the multiple tasks being handled by the computer.

Function Code 82H: Program Termination — This interrupt is used to identify devices that are in use by a program when that program terminates. The device ID must be in register BX. The intended use for this interrupt is to automatically deallocate devices that are no longer in use, similar to the closing of open files at the end of a BASIC program.

Function Code 83H: Event Wait — This function enables a timer-based delay. After the specified number of microseconds has elapsed (this number is loaded in registers CX:DX), the high order bit of the flag byte pointed to by the address in ES:BX will be set to a 1.

NOTE: The firmware does not wait for the time to elapse before returning control to the calling routine. Instead, control is returned immediately. It is up to the program to monitor the flag byte to determine when it has been set by the function.

Function Code 84H: Joystick Support — This function provides joystick support through port 201H. When the value in register DX is 0 (zero), the current joystick switch settings are returned in register AL (bits 7 – 4). When the value in DX is 1, the values representing the x-axis and y-axis position of the joystick, shown in Table 7-4, are returned.

Table 7-4: Joystick Values

REGISTER	DESCRIPTION
AX	Joystick #1 x-axis value
BX	Joystick #1 y-axis value
CX	Joystick #2 x-axis value
DX	Joystick #2 y-axis value

Function Code 85H: System Request Key — This interrupt takes place each time the SYS REQ key is pressed or released. The value returned in AL reports the action of the key (pressed or released). If the value in AL is 0, the key was pressed. If the value in AL is 1, the key was released.

Function Code 86H: Wait — This interrupt will cause the program to wait the number of microseconds stored in CX:DX. Control will be returned to the caller only after the indicated time has elapsed.

Function Code 87H: Block Move — This interrupt will cause a specified block of memory to be moved from one location to another. The amount of memory (measured in 16-bit words) is stored in register CX. The six 8-byte global descriptor tables (GDT) are used to describe to source and destination addresses, along with other information. The function of each GDT is described in Table 7-5. The value in the register combination ES:SI points to the beginning of the descriptor block. During the operation, the 80286 is placed in protected mode so that memory locations beyond the 1 megabyte real mode limit can be addressed. The structure of each GDT is described in Table 7-6.

Table 7-5: Block Move Global Descriptor Tables

BYTE OFFSET ADDRESS RANGE	DESCRIPTION
00H – 08H	Required dummy descriptor table (set to zero).
09H – 0FH	Pointer to the beginning of these tables as a data segment (set to zero).
10H – 18H	Source to be moved descriptor table.
19H – 1FH	Destination descriptor table.
20H – 28H	Descriptor table (to be used by function to create a protected code segment — set to zero).
29H – 2FH	Descriptor table (to be used by function to create a protected stack segment — set to zero).

Table 7-6: Global Descriptor Table Structure

BYTES	DESCRIPTION
00H and 01H	Number of bytes to be moved.
02H, 03H, and 04H	The 24-bit physical address of the first byte in LSB form.
05H	Access rights byte.
06H and 07H	Reserved, must be set to zero.

You can find the access rights byte abnd more information on global and local descriptor tables in the iAPX 286 Programmer's Reference Manual, published by Intel Corporation, 3065 Bowers Ave., Santa Clara, CA 95051.

The maximum number of butes that can be moved is 8000H (32K).

Function Code 88H: Determine Extended Memory Size — MFM-200 will return a value in register AX that represents the amount of contiguous memory installed above the 1 megabyte real mode memory limit. The amount of memory is indicated in 1024 byte increments.

Function Code 89H: Processor to Virtual Mode — This function will place the 80286 CPU into its protected mode. ES:SI contains the base address of the global descriptor table to be used; BH contains the base address of the interrupt vector for the master interrupt controller (8259); BL provides the base address of the interrupt vector for the second interrupt controller (8259).

This function expects the address in the registers ES:SI to point to the global descriptor table that will be used by the 80286 while in protected mode. In addition, the function will reprogram the two 8259 interrupt controllers, setting their base interrupt vector values to those stored in BH and BL. The contents of global descriptor table for this function is described in Table 7-7.

NOTE: Once the 80286 is in protected mode, the MFM-200 interrupt routines are no longer addressable. Therefore, the code segment in protected memory must contain all interrupts and routines needed by the program.

Table 7-7: Global Descriptor Tables for Function Code 89H

BYTE OFFSET ADDRESS RANGE	DESCRIPTION
00H – 08H	Required dummy descriptor table (set to zero).
09H – 0FH	Pointer to the beginning of these tables as a data segment (set to zero).
10H – 18H	Pointer to the interrupt descriptor table (IDT).
19H – 1FH	Pointer to the DS (data segment) in protected memory.
20H – 28H	Pointer to the ES (extra segment) in protected memory.
29H – 2FH	Pointer to the SS (stack segment) in protected memory.
30H – 38H	Pointer to the CS (code segment) to which the function will return. Therefore, it should be initialized to the CS of the original program. Upon completion of the function, the control will return to the code segment specified using the IP (instruction pointer) that was placed on the stack when the INT 15, function code 89, was called.
39H – 3FH	Pointer to the CS (code segment) that the function will use while in protected mode.

The format of each descriptor table described in this table is the same as the one described in Table 7-6, which is located in the discussion for function code 87H. The access rights bytes will be different for the CS and IDT entries. You can find the access rights byte and more information on descriptor tables in the iAPX 286 Programmer's Reference Manual, published by Intel Corporation, 3065 Bowers Ave., Santa Clara, CA 95051.

Function Code 90H: Device Busy — Before MFM-200 enters a timing loop waiting for a given device, it will generate this interrupt function for that device. Register AL must provide the device type code. User programs that wish to perform some function during this "busy loop" can take advantage of this interrupt function. Table 7-8 contains the description of type codes.

Table 7-8: Device Type Codes

TYPE CODE OR RANGE	DESCRIPTION
00H – 7FH	These are serially reusable devices. The operating system must provide serial access for these devices which include the following.
00H	Rigid disk access, timeout supported.
01H	Floppy disk access, timeout supported.
02H	Keyboard, no timeout support.
80H – BFH	These are reentrant devices; typically, a network. The registers ES:BX is used to distinguish different I/O calls, which are allowed to be placed simultaneously.
80H	Network, no timeout support. ES:BX must point to the network control block (NCB).
C0H – FFH	These are "wait-only" codes that have no source for the "wait." The timeout length of the wait is the function number. These codes include the following.
FDH	Floppy disk motor startup time.
FEH	Printer response time.

Function Code 91H: Set Interrupt Complete Flag — This function will be performed when a device indicates that a particular command has been completed. Register AL will contain the device type code (as described in Table 7-8) that has completed the command.

Set/Read Time-of-Day (INT 1AH)

The INT 1AH instructions allows the programmer to set and/or read the CMOS clock time and date, and to set the CMOS alarm on or off. Register AH is used to establish the operation and registers CX and DX contain the values to be placed in the 32-bit counter or read from it. Register AL is used as the flag if the value has rolled over to a new day since the last time it was read.

To read the current time-of-day value from the time-of-day word, set register AH to zero and execute the interrupt. Upon return from the call, register CX will contain the high part of the count, and register DX will contain the low. If the count has rolled over, register AL will not contain zero. If AL contains a zero, the value has not rolled over to a new day.

To set the time-of-day value, place the high count in the CX register and the low count in the DX register. Set register AH to 1 and execute the interrupt.

The value in the 32-bit time-of-day word is updated 18.2159 times a second. Therefore, 1 a.m. would be represented by a count of 65,577 and 12 noon would be 786,926. The values returned by INT 1AH would be as follows: for 1 a.m., register CX would contain 1 and DX would contain 41 (29H); for 12 noon, register CX would contain 12 (0CH) and DX would contain 494 (1EEH). The contents of register AL would depend upon whether the count had rolled over since the last time-of-day value was read. If AL is zero, then the time-of-day has not advanced past 24 hours. If AL is other than zero, it has.

Timer Tick (INT 1CH)

The INT 1CH interrupt provides a means for you to provide timing loops in your programs. The speed of the CPU in PC-compatible computers is not always consistent from one machine to another and, in some cases, computers such as the Z-130, Z-140, and Z-150 PC Series offer switchable-speed CPU clocks. Therefore, this interrupt provides a convenient means of controlling timing loops independent of the speed of the CPU.

This interrupt is called automatically every time INT 08H (the time-of-day interrupt) is executed (18.2159 times a second). Therefore, if you use this interrupt in your program, you must save the registers at the beginning of the routine and then later restore them before returning by way of an IRET instruction.

At power-up, this interrupt points to an IRET instruction. If you find any other routine, it will have been initialized by an application program.

Real-Time Clock (INT 70H)

The INT 70H interrupt is initiated by a hardware interrupt request (IRQ8) or when an alarm from the real-time clock takes place. The alarm is set and handled through INT 15H, function 83H (event wait) and function 86H (wait). The hardware interrupt request takes place approximately 1,024 times a second. Since this interrupt is essentially controlled by hardware, it should not be called by user programs.

Software Redirect of IRQ2 (INT 71H)

The INT 71H interrupt is used in the Z-200 PC Series to redirect hardware interrupt IRQ2 to either INT 02H or INT 75H, depending upon the device that generated the hardware interrupt. If the interrupt was generated by the numeric processor extension (the 80287), then INT 71H will call INT 75H. Otherwise, INT 02H will be called. This interrupt is hardware driven and should not be called by user software.

80287 Non-Maskable Interrupt (INT 75H)

The INT 75H interrupt is called whenever the 80287 numeric processor extension calls for a non-maskable interrupt. It is generated by hardware and therefore should not be used by software routines.

Programming Sound

The Z-200 PC Series computers do not contain dedicated sound chips. However, tones may be generated and played through the speaker via the 8254 programmable peripheral interface chip in three different ways, which may be implemented simultaneously.

- Generate a pulse train by toggling a program control register bit.
- Program the output of channel 2 of the timer/counter to deliver a sound waveform to the speaker.
- Modulate the clock input to the timer/counter through a program-controlled input/output register bit.

Programming sound, particularly music, is a specialized application in itself and, therefore, this capability is usually reserved for specific application programs and high level languages, such as BASIC.

User Memory

In general, programming user memory is straight forward. Within either real or protected memory, most operations can be performed directly from assembly or machine language programs. Typical memory operations include the following.

- Reading and writing data to specific memory locations.
- Allocating and/or reserving specific locations in memory for specific purposes, such as for software interrupt routines or storing user-defined character fonts.
- Storing character strings and numeric values that are widely used by a number of applications.
- Rerouting interrupts to user-defined routines.
- CPU stack operations.
- Moving memory contents from one area to another.
- Using the contents of RAM to control video graphics.

In addition to normal RAM operation, you may disable parity generation and/or checking, either to check the system or for any other purpose you might have that does not require or need memory parity checking.

Memory Address Format

The 80286 uses a 2-part hexadecimal number to designate any specific memory location. It consists of a 4-digit number to identify the segment address and a 4-digit number to identify the offset address within that segment. The format for this value is:

XXXX:YYYY

The first four digits actually represent a 5-digit hexadecimal number, since an imaginary shift left (multiply by 16) is performed on the value to arrive at the RAM bank and row to select.

The second value selects the RAM column from the selected bank. For example, the value 3F3F:5B11 would represent memory address 44F01H (282,369 decimal). 3F3F shifted left equals 3F3F0H (259,056 decimal). 5B11 (23313 decimal) is the offset that is added to the segment address. The result of adding 3F3F0H and 5B11H is 44F01H.

Since the least-significant digit of the segment is always 0, more than one combination of segment and offset can point to the same memory address. For instance, the address in the previous example could also be defined by 3F00:5F01.

The highest and lowest usable segment values are determined by the memory location being defined. In the example address, the lowest usable segment value would be 34F1H, since anything lower would result in an offset greater than FFFFH. The highest usable segment value would be 44F0H.

Keyboard Interrupts

Programming Keyboard Interrupts

The three interrupts that are applicable to the keyboard are described in Table 8-1. The ASCII and key codes for the keyboard are located in Chapter 15, "Keyboard" in Part IV of this manual.

Table 8-1: Keyboard Interrupts

INTERRUPT	FUNCTION
09H	Key Pressed
16H	Keyboard Input/Output
1BH	Keyboard Break

Key Pressed (INT 09H)

The INT 09H instruction is executed each time a key is pressed or released. It reads the key from the keyboard register and encodes it or takes special action if needed (notes the action of shift or control keys). Valid key codes are then placed in the keyboard buffer where they may be acted upon by other software.

You should not change the action of this interrupt routine since it affects input and output from the keyboard.

Keyboard Input/Output (INT 16H)

The INT 16H instruction is widely used to perform a number of keyboard operations. Three function codes (0, 1, and 2) perform key code retrieval (receive a character from the keyboard), check the keyboard buffer to see if any codes are in it, and return the status of the keyboard shift and control keys.

Function Code 00H: Get Character — This function code will cause the interrupt to get a character from the keyboard buffer. If a key has been typed and is waiting in the keyboard buffer, it will be retrieved and removed from the buffer. The code (ASCII codes 00H – 7FH or non-ASCII codes 80H – FFH) assigned to that character will be returned in the AL register. The scan code (01H – 84H) for the key pressed will be returned in register AH. If the keyboard buffer is empty, the routine will wait until a key has been pressed on the keyboard, generating a character code. Note that the shift and control keys do not generate codes, but affect the codes generated by other keys.

Function Code 01H: Check Keyboard Buffer — This function code will cause the interrupt to check the status of the keyboard buffer. If the buffer is empty, the zero flag (ZF) will be set. If the buffer contains key codes waiting to be processed, the zero flag will be clear (not set). The code (ASCII codes 00H – 7FH or non-ASCII codes 80H – FFH) assigned to the first character in the buffer will be returned in the AL register. The scan code (01H – 84H) for that key will be returned in register AH.

NOTE: This operation does not remove any codes from the keyboard buffer. Therefore, if you execute function code 01H followed by function code 00H, the same key codes will be returned. Only function code 00H removes the key codes from the buffer.

Function Code 02H: Get Keyboard Status — This function code will cause the interrupt to report the status of the shift and control keys. The value returned in register AL is defined in Table 8-2.

Table 8-2: Keyboard Status Report

BIT (AL)	DESCRIPTION
0	The right SHIFT key is pressed.
1	The left SHIFT key is pressed.
2	The CTRL key is pressed.
3	The ALT key is pressed.
4	The scroll lock function is active.
5	The number lock function is active.
6	The caps lock function is active.
7	The insert mode is active.

Keyboard Break (INT 1BH)

The INT 1BH instruction is executed when the key pressed interrupt (09H) detects the CTRL and BREAK keys (CTRL-BREAK) at the same time (The CTRL key must be pressed first and then held while the SCROLL LCK/ BREAK key is pressed).

The interrupt for this routine must return with an IRET instruction to properly exit the 09H interrupt. When the computer is turned on, this routine provides only the IRET instruction to make sure that nothing will happen if CTRL-BREAK is pressed during the self-tests.

GW-BASIC is an example of software that uses this routine. It uses this interrupt to halt execution of a BASIC program when the CTRL-BREAK key is pressed.

If you allow this interrupt to retain control, you may have to accommodate one or more of the following conditions.

- The break may occur during interrupt processing. You must then send one or more end-of-interrupt commands to the 8259 controller.
- If an operation was in process when you caused the interrupt to take place, all input/output devices must be reset.
- Programs that use this interrupt must not chain to the previous owner of this interrupt, and the interrupt must be restored when you are finished.

Remember, when servicing an INT 1BH instruction, your routine must perform an IRET to make sure that the interrupt is restored. Do not link to the next routine.

Chapter 9

I/O Interrupts

Programming Input/Output Interrupts

Those interrupts that are generally considered to be input/output interrupts are defined in Table 9-1. For information pertaining to the use and programming of interrupts, see Chapter 6, "Software Interface."

Table 9-1: I/O Interrupts

INTERRUPT	FUNCTION
05H	Print Screen
0BH	Communications (COM2)
0CH	Communications (COM1)
0DH	Alternate Parallel Printer (LPT2)
0FH	Parallel Printer (LPT1)
14H	Serial Input/Output
17H	Printer Input/Output
18H	Parallel/Serial Configuration

Print Screen (INT 05H)

This interrupt sends the contents of the screen to the LPT device (usually a printer). It will only print out valid characters. Graphics that do not match a valid character shape will be ignored. INT 05H performs exactly the same function as the SHIFT-PRTSC entry.

To aid in using the Print Screen function, a byte of global RAM at location 0050:0000 has been reserved as a status byte. While the print screen routine is executing, this byte is set to 1 (this is actually a flag used by the print screen routine to prevent additional print screen requests from being honored while it is executing). Upon completion of the screen dump, the status byte will be 0H if no errors occurred or 0FFH if an error was encountered (usually this is caused by a printer time-out).

Communications (INT 0BH and INT 0CH)

The INT 0BH and INT 0CH instructions were written for serial communications through COM1 (INT 0CH) and COM2 (INT 0BH). However, these interrupts are not normally used by MFM-200 or the MS-DOS operating system. Therefore, if you wish to use these instructions for interrupt-driven input/output activities, you must supply your own input/output code for the devices. Note that INT 14H and INT 18H provide better support for serial communications.

Parallel Printer (INT 0DH and INT 0FH)

The INT 0FH (LPT1) and INT 0DH (LPT2) interrupts are used for parallel communications, usually with printers. However, these interrupts are not normally used by MFM-200 or the MS-DOS operating system. If you wish to use these instructions for interrupt-driven input/output activities, you must supply your own input/output code for the particular device. Note that INT 17H provides better support for parallel communications.

Serial Input/Output (INT 14H)

The INT 14H instruction will allow you to perform the serial functions described in Table 9-2. Each function code is described in the following paragraphs. Also refer to the Parallel/Serial Configuration section in this chapter.

Before the interrupt can be successfully executed, register AH must be loaded with the specified function code. Register DX must be loaded with the serial device designator, 0 or 1 (for COM1 or COM2). Note that the specified serial device must be addressable by the system.

Table 9-2: Serial Device Function Codes

CODE	DESCRIPTION
00H	Initialize the serial input/output port.
01H	Send character in AL to the serial port.
02H	Receive character in AL from serial port.
03H	Read communications status.

Function Code 00H: Initialize the Serial Input/Output Port — This function code allows the interrupt to determine the mode of the selected port by the bit-mapped value contained in AL and described in Table 9-3.

Table 9-3: Mode-Select Byte Breakdown

BIT	DESCRIPTION
0	With bit 1, sets word length (see Table 9-4).
1	With bit 0 sets word length (see Table 9-4).
2	Number of stop bits. Set to 0 for 1 stop bit, or 1 for 2 stop bits.
3	With bit 4, sets parity selection (see Table 9-5).
4	With bit 3, sets parity selection (see Table 9-5).
5	With bits 6 and 7, sets baud rate (see Table 9-6).
6	With bits 5 and 7, sets baud rate (see Table 9-6).
7	With bits 5 and 6, sets baud rate (see Table 9-6).

Table 9-4: Word Length Selection

BIT 1	BIT 0	WORD LENGTH
0	0	Don't care.
0	1	Don't care.
1	0	7 bits.
1	1	8 bits.

Table 9-5: Parity Selection

BIT 4	BIT 3	SELECTION
0	0	No parity.
0	1	Odd parity.
1	0	Don't care.
1	1	Even parity.

Table 9-6: Baud Rate Selection

BIT 7	BIT 6	BIT 5	BAUD RATE
0	0	0	110
0	0	1	150
0	1	0	300
0	1	1	600
1	0	0	1200
1	0	1	2400
1	1	0	4800
1	1	1	9600

Upon return from the initialization function, the status of the serial port will be in AX; AH will contain the line control status and AL the modem control status. See the Function Code 03H: Read Communications Status description for details.

Function Code 01H: Send Character to the Serial Port — This function code will cause the interrupt to transmit a byte out the serial port. Upon return from this call, the most-significant bit of register AH will contain the transmit status. If the routine could not transmit the character placed in AL, the transmit status will be set to 1. The remaining AX bits will reflect the status as defined in the discussion of Function Code 03H: Read Communications Status.

Function Code 02H: Receive Character from the Serial Port — This function code will cause the interrupt to receive a byte from the serial port. Upon return from this call bits 7, 4, 3, 2, and 1 of AH will contain the data transfer status. Function Code 03H: Read Communication Status describes the transfer status information. If register AH is set to 0, the routine has read the byte properly into AL. If AH is not 0, some type of error occurred. In these cases, timeout errors refer to either the absence of the data set ready (DSR) or clear to send (CTS) signals.

Function Code 03H: Read Communications Status — This function code will cause the interrupt to check the communications interface status. The status is returned as a bit-mapped value in register AX. Register AH contains the line control status and register AL contains the modem control status, as shown in Tables 9-7 and 9-8.

Table 9-7: Line Control Status (Register AH)

BIT	STATUS
0	Received data ready.
1	Overrun error. The CPU did not read the previous character and a new character has arrived.
2	Parity error. Parity was enabled, but the parity of the incoming character did not match the programmed value.
3	Framing error. An incorrect start/stop bit was received.
4	Break detect.
5	Transmitter holding register empty. Serial input/output channel is ready for another character to transmit.
6	Transmitter shift register empty. The serial input/output channel is not currently transmitting.
7	Timeout error. Device did not respond within a reasonable period of time.

Table 9-8: Modem Control Status (Register AL)

Bit	Status
0	Clear to send (CTS) line has changed state.
1	Data set ready (DSR) status has changed.
2	End of ringing pulse detector.
3	Carrier detect (CD) signal has changed state.
4	Status of CTS line.
5	Status of DSR line.
6	Ringing indicator.
7	Carrier detect status.

Printer Input/Output (INT 17H)

The INT 17H instruction is used to perform input/output functions with the parallel port on the computer. The configuration table (as described in the Parallel/Serial Configuration section of this chapter) is referenced by this interrupt for proper operation. Table 9-9 describes the function codes of this interrupt.

Table 9-9: Parallel Device Function Codes

FUNCTION CODE	DESCRIPTION
00H	Print the next character in AL. If the parallel device does not return a ready status within a reasonable amount of time, bit 0 of register AH will be set to 1, otherwise the register will return the status, as described in function code 2. Register DX must contain the port to be used (00H – 03H).
01H	Initialize the parallel port. Register AH will contain the device status following execution of the interrupt; Refer to function code 2.
02H	This function will cause the interrupt to return the status of the device in register AH as follows. Bit 0 — Time out error, device not ready. Bit 1 — Not used. Bit 2 — Not used. Bit 3 — Input/output error. Bit 4 — Device on-line. Bit 5 — Out of paper signal. Bit 6 — Character acknowledge. Bit 7 — Device is busy or in an error state.

Parallel/Serial Configuration (INT 18H)

The INT 18H instruction is used to point to the parallel and serial device tables that can be used to configure your computer's features. Most important of these is the ability to divert the parallel output to a serial output channel. For more information on configuration from the operating system, see the owner's manual for your computer. Refer to Related Publications in Chapter 1, "Introduction."

NOTE: This interrupt is used to point to a resident BASIC in ROM in PC-compatible computers.

In order to maintain compatibility, the default configuration table values are loaded from the MFM-200. Application programs can install another pointer to new tables, if needed, to implement specific features. The MS-DOS CONFIGUR utility modifies the parameters in these tables as described in the documentation for the operating system as well as the owner's manual.

Figure 9-1 illustrates the format for the serial and parallel device parameter mapping table. Up to three parallel devices, and two serial devices, may be configured. In addition, a parallel device may be mapped to a serial device.

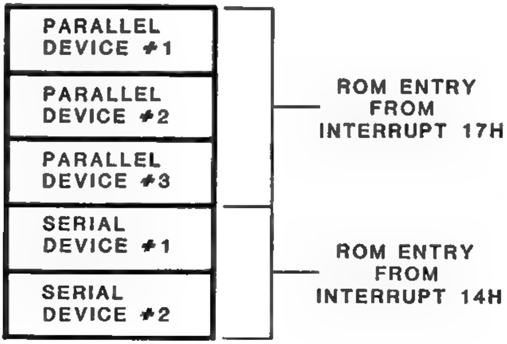


Figure 9-1: Serial and Parallel Device Layout

Parallel Format

The format for each of the three parallel maps is shown in Table 9-10.

Table 9-10: Parallel Map Format

BYTE	PARAMETER DESCRIPTION
1	Performs remapping, parity, and upper-/lowercase handling as follows. Bits 0 – 3: 00 — No remapping. 01 — Remap parallel output to COM1. 10 — Remap parallel output to COM2. 11 — Not used. Bit 4: Strip parity on output. Bit 5: Not used. Bit 6: Map lowercase output to uppercase. Bit 7: Not used.
2	The pad character to be used after a CR.
3	Number of pad characters to be sent.
4	ROM data segment time out value for a parallel device.

Serial Format

The format for the two serial maps is shown in Table 9-11.

Table 9-11: Serial Map Format

BYTE	PARAMETER DESCRIPTION
1	Type of handshake used by device (see Table 9-12).
2	Attributes of transmission (see Table 9-13).
3	Pad character after carriage return.
4	Number of pad characters to issue.
5	Flag for burst count for ETX/ACK.
6	Counter for ETX/ACK.
7	Device initialization (see Table 9-14).
8	Reserved.

The serial byte #1, serial byte #2, and serial byte #7 breakdowns are detailed in Tables 9-12, 9-13, and 9-14 respectively.

Table 9-12: Serial Byte #1 Breakdown

BIT	PARAMETER DESCRIPTION
0	Compatibility bit. If logic 1, IBM compatible DTR and RTS high. If logic 0, handshake determined by bit 1.
1	Protocol bit. If logic 0, hardware handshake. If logic 1, software handshake.
2	If bit 1 is logic 0, bit 2 is logic 1 if DTR handshake active, logic 0 if not. If bit 1 is logic 1, bit 2 is logic 1 if DC1/DC3 handshake, logic 0 if ETX/ACK handshake.
3	If bit 1 is logic 0, bit 3 is polarity of DTR if DTR active. If bit 1 is logic 1, bit 3 is logic 1 if waiting for handshake character, logic 0, if not.
4	If bit 1 is 0, bit 4 is logic 1 if RTS handshake active, 0 if not.
5	If bit 1 is logic 0, bit 5 is polarity of RTS if RTS active, 0 if low, 1 if high.

Table 9-13: Serial Byte #2 Breakdown

BIT	PARAMETER DESCRIPTION
0	If logic 0, do not strip parity on input; if logic 1, strip parity.
1	If logic 0, do not strip parity on output; if logic 1, strip parity.
2	If logic 1, map lowercase to uppercase on input; if logic 0, do not.
3	If logic 1, map lowercase to uppercase on output; if logic 0, do not.

Table 9-14: Serial Byte #7 Breakdown

BIT	PARAMETER DESCRIPTION
0, 1	Word size.
2	Number of stop bits.
3, 4	Define parity; yes/no odd/even.
5, 6, and 7	Baud rate (see Table 9-6).

Chapter 10

Data Storage Interrupts

Programming Disk Drive Interrupts

Those interrupts that are generally associated with disk drives are defined in Table 10-1. For information pertaining to the use and programming of interrupts, see Chapter 6, "Software Interface."

Table 10-1: Data Storage Interrupts

INTERRUPT	FUNCTION
0EH	Floppy Disk Drive
13H	Disk Input/Output
19H	Booting an Operating System
1EH	Disk Parameters
76H	Rigid Drive Controller

Floppy Disk Drives (INT 0EH)

The INT 0EH instruction is used for communication with floppy disk drives. It is somewhat limited for programming purposes. You will find the INT 13H instruction more useful for most disk communication.

Disk Input/Output (INT 13H)

The INT 13H instruction allows you to transfer information to and from the system's disk drives. Information is provided for floppy disk drive and high-performance Winchester disk drive(s).

Table 10-2 lists the hexadecimal input/output port addresses associated with the disk drives.

Table 10-2: Disk Drive Input/Output Ports

PORT	REGISTER
3F2H	Floppy disk digital output.
3F4H	Floppy disk controller status.
3F5H	Floppy disk controller data.
320H	Winchester drive #1.
322H	Winchester drive #2.

To access a particular drive, you must use one of the drive codes defined in Table 10-3 to identify it.

Table 10-3: Drive Identification Codes

CODE	DRIVE
00H	Floppy disk drive 1.
01H	Floppy disk drive 2.
02H	Floppy disk drive 3.
03H	Floppy disk drive 4.
80H	Winchester disk drive 1.
81H	Winchester disk drive 2.
82H	Winchester disk drive 3.
83H	Winchester disk drive 4.

The INT 13H instruction uses the value in register AH to determine the type of function performed by the interrupt. They are defined in Table 10-4 and described in the following discussion.

Table 10-4: Disk Drive Function Codes

CODE	FUNCTION
00H	Reset disk system.
01H	Read disk status.
02H	Read specified sectors into a buffer.
03H	Write buffer contents to desired sectors.
04H	Verify that selected sectors can be read without error. No information is actually transferred using this call.
05H	Format track.
06H	Flag selected track as a bad track.
07H	Format drive starting with specified track.
08H	Return current drive parameters.
09H	Initialize drive type characteristics.
0AH	Read sectors including 4 ECC bytes at end of each sector.
0BH	Write sectors including ECC bytes.
0CH	Seek a desired track.
0DH	Reset Winchester controller only.
0EH	Read contents of controller's sector buffer into user-specified buffer.
0FH	Write user's buffer into sector buffer.
10H	Test to see if specified drive is ready.
11H	Recalibrate drive to track 0.
12H	Execute controller RAM diagnostics.
13H	Execute drive test diagnostics.
14H	Execute internal controller diagnostics.
15H	Read drive type.
16H	Read floppy drive change line.
17H	Set type for format.

NOTE: Codes 06H through 14H apply only to rigid disk (Winchester) drives.

Function Code 00H: Reset Disk System — This function code will cause the interrupt to reset each disk in the system (move the read/write heads to track 0). For floppy disk drives, interrupt vector 1EH should point to a drive parameter block that contains the bytes defined in Table 10-5.

Table 10-5: Drive Parameter Block

BYTE	PARAMETER
00H	Disk controller command byte 1.
01H	Disk controller command byte 2.
02H	Number of clock ticks until motor is turned off following disk input/output
03H	Bytes per sector: 0 = 128, 1 = 256, 2 = 512, and 3 = 1024.
04H	Last sector of track.
05H	Gap length in bytes.
06H	Data length: 80H if 128 bytes per sector, FFH if not.
07H	Gap length when formatting.
08H	Fill data used during formatting.
09H	Head settling time (milliseconds).
0AH	Motor start time (.25 second increments).

Function Code 01H: Read Disk Status — This function code will cause the interrupt to return the disk status in AL. This call is used when an applications program requires to see the disk status generated by the last disk input/output operation. The status will be the same as that returned by the last input/output call (see function codes 2 – 5).

Table 10-6 defines the parameters required by function codes 02H – 05H.

Table 10-6: Required Parameters for Function Codes 02H – 05H

REGISTER	PARAMETER
DL	Drive ID code as previously defined.
DH	Head (side) number. 0 or 1 for floppy disk drives. 0 – 7 for Winchester drives.
CH	Track number.
CL	Sector number. On Winchester input/output calls, the two MSBs of CL contain the two MSBs of the track number
AL	Number of sectors to be transferred.
ES:BX	Pointer to the disk buffer; segment in ES. Offset in BX not required for function code 4.

Function Code 02H: Read Sector(s) — This function code will cause the interrupt routine to read the specified number of sectors into the disk buffer (see Table 10-6).

Function Code 03H: Write Sector(s) — This function code will cause the interrupt routine to write the contents of the disk buffer into the specified number of sectors on the disk (see Table 10-6).

Function Code 04H: Verify Sector(s) — This function code will cause the interrupt routine to test the specified sector(s) to see if they can be read without any errors. No data is actually transferred (see Table 10-6).

Function Code 05H: Format a Track — This function code will cause the interrupt routine to format a track. The specified track information must be placed in sector headers in memory. The buffer pointer, ES:BX, must point to the beginning of that memory. Each sector on the track must have a sector header and each header must contain four bytes of information in the following order: track number, head (disk side) number, sector number, and bytes per sector (refer to function code 00H). Place the number of sectors per track in register AL before you execute the interrupt instruction. The data stored in the disk buffer will be used as a filler for each sector's data field.

If the drive is a Winchester, you must first load the data to be written to the disk into the controller's sector buffer using Function Code 0FH.

NOTE: The following function codes are used only for Winchester disk input/output. Table 10-7 describes the contents of the CPU registers prior to executing function codes 06H, 07H, and 0CH.

Table 10-7: Register Requirements for Function Codes 06H, 07H, and 0CH

REGISTER	CONTENTS
AH	Function code number (06H, 07H, or 0CH).
AL	Interleave factor (01H – 10H) — not required for 0CH.
CH	Cylinder number (00H – 03FFH).
CL	Sector number to read. NOTE: The 2 most-significant bits of the cylinder number are also placed in the 2 most-significant bits of this register.
DH	Drive head number (00H – 07H).
DL	Drive number (80H – 87H).

Function Code 06H: Flag Bad Track — This function code will cause the interrupt routine to flag the specified track (determined from the head number and cylinder number) as a bad track.

Function Code 07H: Format Drive Starting at Specified Track — This function code will cause the interrupt routine to format the remainder of the drive, starting at the specified track (determined from the head number and cylinder number). The data in the disk buffer will be used as a filler in the data field in each sector.

Function Code 08H: Return Current Drive Parameters — When completed, register DH will contain the number of active Winchester drives, register DL will contain the maximum number of usable heads, register CH will contain the maximum number of usable cylinders, and register CL will contain the maximum usable sector number with the two most significant bits representing the maximum usable cylinder number.

Function Code 09H: Initialize Drive Type Characteristics — This function uses the parameters pointed to by interrupts 41H, 46H and (if three drives are supported) 4BH to initialize the controller for the specific drive types. Refer to Table 10-8.

Table 10-8: Drive Type Characteristics

DATA TYPE	DESCRIPTION
Word	Maximum number of cylinders.
Byte	Maximum number of heads.
Word	Reserved.
Word	Start cylinder for write precompensation.
Byte	Maximum number of bits to correct on data (ECC) errors.
Byte	Controller flag options. If bit 7 or 6 is 1, retries are disabled. If bit 3 is 1, it indicates a drive containing more than eight heads.
3 Bytes	Reserved.
Word	Landing zone.
Byte	Number of sectors/track.
Byte	Reserved.

Function Code 0AH: Read Long — This function code is identical to function code 02H with the exception that the ECC correction is enabled. Error 11H is generated when an ECC recoverable error is detected. The ECC algorithm has attempted to correct the data in the buffer. Refer to Table 10-9 for register content requirements.

Function Code 0BH: Read Long — This function code is identical to function code 03H with the exception that the ECC correction is enabled. Error 11H is generated when an ECC recoverable error is detected. The ECC algorithm has attempted to correct the data in the buffer. Refer to Table 10-9 for register content requirements.

Table 10-9: Register Requirements for Function Codes 0AH and 0BH

REGISTER	CONTENTS
AH	Function code (0AH or 0BH).
AL	Number of sectors to be read (01H – 4FH).
ES:BX	Address of disk buffer in memory.
CH	Cylinder number (00H – 3FFH).
CL	Sector number to be read. **NOTE:** The 2 most-significant bits of the cylinder number are placed in the 2 most-significant bits of the CL register.
DH	Head number (00H – 07H).
DL	Drive number (80H – 87H).

Function Code 0CH: Seek Track — The read/write heads of the drive whose number is in DL (80H is the first Winchester, 81H is the second, etc.) is moved to the cylinder indicated by CH. Valid cylinder numbers are 0 – 1023. Refer to Table 10-7 for the CPU register content requirements.

Function Code 0DH: Alternate Disk Reset — This function code will cause the interrupt routine to reset only the Winchester controller values to default for the specified drive. Register DL must contain the drive number (80H – 87H). The normal disk reset function (00H) also resets the floppy disk subsystem.

Function Code 0EH: Read Sector Buffer — This function code will cause the interrupt routine to read the controller's sector buffer for the specified drive into memory. Register AL must be set to 1, ES:BX is the starting address in memory to which the sector buffer data will be transferred, and DL must contain the drive number (80H – 87H).

Function Code 0FH: Write Sector Buffer — This function code will cause the interrupt routine to write the contents of memory into the controller's sector buffer for the specified drive. Register AL must be set to 1, ES:BX is the starting address in memory from which the data will be transferred, and DL must contain the drive number (80H – 87H).

Function Code 10H: Test Drive Ready — This function code will cause the interrupt routine to check the status of the specified drive. Register DL must contain the drive number to be tested (80H – 87H). The status is returned in register AH. If the drive is ready, the status will be 00H.

Function Code 11H: Recalibrate — This function code will cause the interrupt routine to recalibrate the specified drive (move the read/write heads to cylinder 0. Register DL must contain the drive number (80H – 87H).

Function Code 12H: Execute Controller Memory Diagnostic — This function code will cause the interrupt routine to run a data-pattern test on the controller's RAM buffer for the drive specified. Register DL must contain the drive number (80H – 87H).

Function Code 13H: Execute Internal Diagnostics — This function code will cause the interrupt routine to run a test of the specified drive and drive-to- controller interface. It consists of a recalibrate command, followed by a series of seek commands. Sector 0 of head 0 is tested to see if it can be read successfully. These tests are nondestructive; no write commands are performed. Register DL must contain the specified drive number (80H – 87H).

Function Code 14H: Controller Self-Test — This function causes the controller to initiate an internal diagnostic sequence. On return, AH will equal 0 if the controller passes the tests. If the controller fails the tests, the value in AH will represent an error code.

Function Code 15H: Read Drive Type — On entry, register DL must contain the number of the drive whose type is being requested. The first drive is number 0, the second drive is number 1, etc. On return, AH will contain the following information.

- When the value in AH is 0, it indicates that a drive does not exist.
- When the value in AH is 1, it indicates that a floppy drive exists with no disk change line.
- When the value in AH is 2, it indicates that a floppy drive exists that does have a disk change line.
- When the value in AH is 3, it indicates that a Winchester disk drive is present.

The Winchester version of this function performs in an identical fashion with the following exception. CX:DX will indicate the number of 512 byte blocks available on the disk.

Function Code 16H: Read Floppy Disk Change Line — This function allows user programs to check the status of floppy disk drives that use the "disk change line" (refer to Function Code 15H). On entry, DL contains the drive number of the drive to be interrogated. On exit, the CY flag, and register AH report the status of the disk change line as follows.

- When the value in register AH is 0, it indicates that a disk has not been changed.
- When the value in register AH is 6, and the CY flag is set, it indicates that a disk has been changed.

Function Code 17H: Set Type For Format — This function allows the user to inform MFM-200 of the type of floppy disk, and floppy disk drive to assume when issuing a subsequent format command (Function 05H). Register DL must contain the number of the drive that is to be set and register AL must contain the disk/drive type as follows.

- When the value in register AL is 1, it indicates that a 360K disk and 360K drive are being used.

- When the value in register AL is 2, it indicates that a 360K disk and a 1.2M drive are being used.
- When the value in register AL is 3, it indicates that a 1.2M disk and 1.2M drive are being used.

NOTE: The value of 0 (zero) is not used in register AL to define a drive type.

Whenever the value in register AL is 2 or 3, the disk change line is interrogated before the drive type is set. If the disk change line is found to be inactive, the drive type is set and no error is generated.

If the disk change line is found to be active, an attempt is made to deactivate it. When the disk change line is made inactive, the drive type is set and a disk change error is generated. If for any reason the disk change line cannot be made inactive, a timeout error is generated and the drive type is set to an unknown state.

Error Status Codes

All of the drive function calls will return a status code in register AH upon completion. Register AH and the carry flag CF will contain a 0 if the function was successful. If the function failed, the carry flag will be set and register AH will contain a value representing a specific error. Table 10-10 defines the error codes.

Table 10-10: Disk Drive Error Codes

CODE	ERROR
00H	No error (see text).
01H	Bad command. An invalid command was provided in the parameters of the interrupt.
02H	Bad address mark. The disk controller could not find an address mark.
03H	Write protect. Occurs if a write or format operation was attempted and the disk was write protected.
04H	Record not found. The controller could not find the requested sector.

Table 10-10 (continued): **Disk Drive Error Codes**

CODE	ERROR
05H	Winchester controller reset failed.
07H	Controller would not accept drive parameters.
08H	DMA overflow. This occurs when the DMA controller on the I/O board cannot keep up with the disk data transfer, and information is lost. It usually results from excessive DMA action from other devices on the data bus or hardware failure of the bus interface.
09H	DMA boundary error. The hardware is incapable of transferring sector buffers information across 64K memory boundaries. This error can be avoided by reducing the sector count in register AL or by changing buffer pointer so that entire transfer area resides within a single 64K memory segment.
10H	Bad CRC on disk read. Specified record was found, but the cyclic redundancy check (CRC) for the data did not match the value calculated by the controller. On Winchester drives, this error flags any error that the ECC circuitry could not correct.
11H	On Winchester drives, this code indicates that an ECC error has occurred, but the controller was unable to reconstruct the lost data.
20H	Disk controller IC has failed.
40H	Bad seek. The disk controller attempted to move the read/write head to a specified track, but could not find a matching sector header on that track.
80H	Timeout error. This occurs when a command has been issued to the controller, but was not completed within an amount of time specified by the hardware.
BBH	Undefined error. this usually results from a bad controller IC.
FFH	On Winchester drives, this indicates that a sense drive status operation has failed.

Booting an Operating System (INT 19H)

The INT 19H instruction will attempt to boot an operating system from the specified disk drive. Register DL must contain the drive number and if it is the Winchester disk drive, AL must contain a partition number between 0 and 3 expressed in ASCII format (that is, a value between 30H to 33H). If the value is 0, the boot track of the default partition will be selected.

The routine will attempt to read from track 0, sector 1 of the specified device, and execute the code once it is located and successfully loaded into memory. If the boot routine fails because the drive did not exist or because of a hardware failure, an error message is displayed, and control is passed to the monitor ROM.

Disk Parameters (INT 1EH)

The INT 1EH instruction points to an area in memory that contains the disk parameters for the system. It is initialized when the system is turned on to provide support for loading the operating system from disk. The data stored in this area of memory is described in Table 10-11. The tables will have to be modified if you use different disk drives than those supplied by Zenith Data Systems.

Table 10-11: Disk Parameter Table

BYTE	DESCRIPTION
1	Bits 0 – 3 contain the head unload time. The range is 16 to 240 milliseconds in 16-millisecond increments. Bits 4 – 7 contain the head step rate for the drive. The range is 1 to 16 milliseconds in 1-millisecond increments, in reverse order — 0FH = 1 millisecond, 0EH = 2 milliseconds, and so on.
2	Bits 1 – 7 contain the head load time (the range is 2 to 254 milliseconds in 2-millisecond increments). Bit 0 contains the DMA flag. If bit 0 = 0, the DMA mode will be used.
3	Motor wait time. This is the amount of time the motor will remain on after the last disk operation. It is measured in timer ticks.
4	Data bytes per sector. See Table 10-12.
5	Sectors per track.
6	Gap length of gap 3 for read/write operations. The value of this byte is determined by the value in byte 4. See Table 10-12.
7	If byte 4 (sector length) is 0, then this value is the length of the data that will be read from or written to each sector. If byte 4 is not 0, then set this value to FFH; it is ignored by the controller.
8	Gap length of gap 3 for format operations. The value of this byte is determined by the value in byte 4. See Table 10-12.
9	Format data pattern. This is the pattern used by the format command to fill and test the sector.
10	Head settle time measured in 1-millisecond increments.
11	Motor start time measured in 125-millisecond increments.

Table 10-12: **Sector and Gap Lengths**

BYTE 4 VALUE	BYTES PER SECTOR	SECTORS PER TRACK	WRITE GAP LENGTH	FORMAT GAP LENGTH
FM Mode				
00H	128	12H	07H	09H
00H	128	10H	10H	19H
01H	256	08H	18H	30H
02H	512	04H	46H	87H
03H	1024	02H	C8H	FFH
04H	2048	01H	C8H	FFH
MFM Mode				
01H	256	12H	0AH	0CH
01H	256	10H	20H	32H
02H	512	08H	2AH	50H
03H	1024	04H	80H	F0H
04H	2048	02H	C8H	FFH
05H	4096	01H	C8H	FFH

NOTE: MS-DOS version 3 uses MFM Mode, 512 bytes per sector, and 9 sectors per track. The UCSD p-system can use MFM mode, 512 bytes per sector, and 10 sectors per track. Neither of these formats is represented in this table.

Rigid Disk Drives (INT 76H)

The INT 76H instruction is used for communication with rigid disk drives. The equivalent interrupt in PC-compatible machines is INT 0DH. Since both are somewhat limited for programming purposes, they are not widely used. You will find the INT 13H instruction more useful for most disk communication.

Chapter 11

Video Interrupts

Programming Video Interrupts

Those interrupts that are generally considered to be video interrupts are defined in Table 11-1. For information pertaining to the use and programming of interrupts, see Chapter 6, "Software Interface."

Table 11-1: Video Interrupts

INTERRUPT	FUNCTION
10H	Video Input/Output
1DH	Video Initialization
1FH	Defining Characters

Video Input/Output (INT 10H)

The INT 10H instruction can be programmed to provide many different input/output functions, based on the value (Function Code in Table 11-2) placed in register AH. The operations performed by these values are defined in Table 11-2. The following paragraphs describe each operation.

Table 11-2: Video Input/Output Function Codes

FUNCTION CODE	REGISTERS AFFECTED	OPERATION
00H	AL	Set screen mode.
01H	CH, CL	Set cursor size.
02H	DH, DL	Set cursor position.
03H	DH, DL, CH, CL	Read cursor position.
04H	AH, DH, DL, CH, BH, BL	Read light pen position.
05H	AL	Select active display page.
06H	AL, BH, CH, CL, DH, DL	Scroll an area of screen up.
07H	AL, BH, CH, CL, DH, DL	Scroll an area of screen down.
08H	AH, AL, BH	Read cursor position.
09H	AL, BH, BL, CH, CL	Send character/attribute to screen.
0AH	AL, BH, CH, CL	Send character to screen.
0BH	BH, BL	Set graphics foreground color.
0CH	AL, CH, CL, DH, DL	Write graphics pixel.
0DH	AL, CH, CL, DH, DL	Read graphics pixel.
0EH	AL, BH, BL	Dumb terminal display.
0FH	AH, AL, BH	Returns current video state.
10H	AL, BL, BH	Set palette registers.
11H	AL,BL	Character generator routine.
12H	BH, BL, CH, CL	Alternate select.
13H	AH, BL	Write string of characters.
100H	AL	Set scroll mode.
101H	AL	Set interlace mode.
102H	AL	Compatibility mode.

Function Code 00H: Set Video Mode — This function code will cause the interrupt to set the video mode. The video mode code, described in Table 11-3, must be placed in register AL. Note that video mode codes 00H – 07H provide standard IBM-compatible video, and that video mode codes 08H – 10H set the video mode for the extended graphics card.

Table 11-3: Video Mode Codes

VIDEO MODE CODE	DESCRIPTION
00H	40 characters by 25 rows, monochrome text display at the RGB output.
01H	40 characters by 25 rows, color text display at the RGB output.
02H	80 characters by 25 rows, monochrome text display at the RGB output. Individual video pages may be scrolled without affecting other video pages.
03H	80 characters by 25 rows, color text display at the RGB output. Individual text pages may be scrolled.
04H	320 × 200 pixel resolution, color graphics at the RGB output. Text emulation capability for 40 characters by 25 rows.
05H	320 × 200 pixel resolution, monochrome graphics at the RGB output. 40 characters by 25 rows, text emulation.
06H	640 × 200 pixel resolution, monochrome graphics at the RGB output. All three scrolling modes available (see scrolling selection).
07H	80 characters by 25 rows, monochrome text display using a monochrome adapter card (such as the Z-329 installed in a PC-compatible connector).
08H	Reserved.
09H	Reserved.
0AH	Reserved.
0BH	Reserved for internal use.
0CH	Reserved for internal use.
0DH	320 × 200 pixel resolution, color graphics at the RGB output. Text emulation for 40 characters by 25 rows.
0EH	640 × 200 pixel resolution, color graphics at the RGB output. Text emulation for 80 characters by 25 rows.
0FH	640 × 350 resolution, graphics display display using a high-resolution monochrome monitor.
10H	640 × 350 resolution, color graphics display using a high-resolution color monitor.

Function Code 01H: Set Cursor Size — This function code will cause the interrupt to set the cursor size. Bits 0 – 4 of register CH must be loaded with the starting scan line number of the cursor. Bits 0 – 4 of register CL must be loaded with the ending scan line number. While it is possible to load any value between 0 and 31, characters in these computers usually contain only eight scan lines, so you should use values in the range 0 – 7 for the starting and ending scan line. Note that the starting scan value may not be greater than the ending scan value.

Function Code 02H: Select Cursor Position — This function code will cause the interrupt to move the cursor to a specified row and column on the screen. Load register DH with the desired cursor row number, and register DL with the column number. When DH is set to 0 and DL is set to 0, the cursor will be placed in the upper left-hand corner of the screen. Load BH with the desired video page number, consistent with the video mode selected (use 0 for graphics mode). Also see information on setting the video page display functions.

Function Code 03H: Read Cursor Position — This function code will cause the interrupt to obtain the current cursor position. Register BH must contain the page number. After the interrupt has been executed, register DH will contain the cursor row and register DL the cursor column position. Register CH and register CL will contain the cursor starting and ending scan line values.

Function Code 04H: Read Light Pen Position — This function code will cause the interrupt to attempt to obtain the light pen position. After the interrupt has been executed, register AH will contain the light pen trigger/switch status: 0 or 1. If AH contains a 0, the light pen has either not been triggered or has not been switched active. If AH is 1, then register DH will contain the row, and register DL the column where the light pen was detected. The scan line number (0 to 199) will be in register CH and register BX will hold the pixel column (0 to 319 or 0 to 639, depending on the graphics mode).

Function Code 05H: Select Active Display Page — This function code will cause the interrupt to select and display the specified page. In any text mode, any unused portion of video memory may be used for additional video pages. In modes 0 and 1, the valid page numbers are 0 to 7; in modes 2 and 3, allowable values are 0 to 3. In the graphics modes, only page 0 is allowed. Load the page number into register AL before calling the interrupt.

Function Code 06H: Scroll an Area of the Screen Up — This function code will cause the interrupt to scroll a specified area of the screen up a specified number of lines. Register CX must be loaded with the upper left-hand coordinates (place the row in register CH and the column in register CL), and register DX with the lower right-hand coordinates. Load the attribute byte for blank lines in register BH. Load register AL with the number of lines to scroll. If register AL is 0, the entire designated window will be cleared.

NOTE: If hardware or smooth scrolling are enabled when this function is implemented, they will be activated by this call if the entire screen is scrolled. It is not possible to use hardware or software scrolling on a portion of the screen.

Function Code 07H: Scroll an Area of the Screen Down — This function code will cause the interrupt to scroll a specified area of the screen down a specified number of lines. Load registers the same as for scrolling up (see Function Code 06H: Scroll an Area of the Screen Up).

Function Code 08H: Read Character and Attribute — This function code will cause the interrupt to return the character and attribute byte codes for the character that resides at the current cursor position. Before calling this routine, load the desired video page in BH for text modes. Upon return from this routine, AL will contain the character code and AH the character attribute code.

Function Code 09H: Write Character and Attribute to Screen — This function code will cause the interrupt to write the specified character and attribute code to the current cursor position. Load BH with the desired page number, CX with a value representing the number of times to repeat the character, AL with the character code, and BL with the attribute code.

Function Code 0AH: Write Character to Screen — This function code will cause the interrupt to write the specified character to the screen; it does not write the attribute to the screen. To use this function, load BH with the page number, CX with the number of characters to write, and AL with the character code.

Function Code 0BH: Select Current Color Palette — This function code is only available in 320 × 200 graphics mode. Load BH with a number from 0 to 127 to represent the identification of the palette. Load the value of the actual color desired (0 to 4) into BL.

If BH is even, the current background color will be used for the foreground graphics color, normally a value between 0 and 31. Values above 15 will select the intensified level of the 0 to 15 colors.

If BH is odd, one of the two available palettes is selected by the value in BL. If BL is 0, the green (1), red (2), yellow (3) palette is selected; if BL is 1 the cyan (1), magenta (2), white (3) palette is selected. The values in parentheses represent the color numbers used by the graphics calls.

Function Code 0CH: Write Graphics Pixel — This function code will cause the interrupt to light a single pixel to a specified graphics coordinate on the current screen page. Load DX with the value of the vertical coordinate (pixel row number in the range 0 – 199) and CX with the value of the horizontal coordinate (pixel column number in the range of 0 – 319 or 0 – 639, depending on whether 320 × 200 or 640 × 200 video mode has been selected). Load AL with the color of the pixel (0 – 3) if in 320 × 200 graphics mode or 0 – 1 if in 640 × 200 mode.

If the most significant bit (bit 7) of AL is set, the color value given will be exclusively oRed with the current color value. This permits simple animation to be implemented.

Function Code 0DH: Read Graphics Pixel — This function code will cause the interrupt to report the color of the pixel in the specified location. The results will be placed in register AL. Before calling, load DX with the vertical (row) pixel position value and CX with the horizontal (column) pixel position.

Function Code 0EH: Dumb Terminal Display — This function code will cause the interrupt to act upon the specified character as if it was being sent to a dumb terminal. That is, the back space (08H), carriage return (0DH), line feed (0AH), and bell (07H) will be treated as console commands rather than characters to be used for screen formatting. Furthermore, if a character occurs at the end of a screen line, the cursor is positioned to the start of the next line. If a line feed is performed in the last display line on the screen, or if a character occurs in the last position of this line, the screen will be scrolled up one line. When scrolling is performed, the attribute for blanked rows in text mode is determined by the attribute at the cursor position on the previous line prior to the scroll.

Load the character to be written in register AL, the foreground color in register BL (for graphics modes), and the display page number in register BH.

Function Code 0FH: Return Current Video State — This function code will cause the interrupt to return the current video status. Register AL will hold the current video mode (see Function Code 00H), register AH the screen width in character columns, and register BH the currently active video page.

Function Code 10H: Set Palette Registers — This function code allows you to set values in specific palette registers. There are 16 separate palette registers that provide selection up to 64 different colors. Note that only 16 of the possible 64 colors may be displayed at any time. The value placed in register AL is used to set the internal palette register parameters.

When the value in register AL is 0, each individual palette register may be set to provide a specific color. The value placed in register BL selects which of the 16 palette registers that will be set, and the value placed in register BH represents the value (color) to be set.

When the value in register AL is 1, the overscan register can be accessed to establish a color that may be used, for example, as a screen border. The value placed in register BH represents the value (color) to be set.

When the value in register AL is 2, the palette register and overscan register may be set simultaneously. ES:DX will point to a 17-byte table with bytes 0-15 representing the palette values, and byte 16 representing the overscan value.

When the value in register AL is 3, the "blink" bit (most significant) used in standard PC-compatible video, may be toggled to represent a second intensity bit. The intensified signal allows 32 different color combinations to be achieved (PC-compatible video allows 16). When the value placed in register BL is 0, the "blink" bit is enabled as a second intensity bit, and when the value in BL is 1, the "blink" bit is not intensified.

Function Code 11H: Character Generator Routine — This function code allows you to establish different character sets for text and graphics modes.

Up to four different character sets can be stored in memory but only two character sets can be accessed at the same time. You can however, switch between the character sets. It should be noted that the extended graphics card must be populated with more than 64K of memory if four character sets are to be used. In addition, register (AL) values of 0, 1, 2, 3, 10, 11, and 12 are used in text modes, and values 20, 21, 22, and 23 are used in graphics modes.

When the value in register AL is 0, you may create your own character set. The placement of the character set (set 1, set 2, etc.) in memory is determined by the value (0-3) in register BL. The value in register BH indicates the number of bytes per character (in an 8 × y format). Note that the maximum value for y is 32 (8 × 32). The value in register CX indicates how many characters will be defined (maximum of 256). The value in register DX indicates the starting character in the set you created. For example, if you specify in register CX that your character set will contain four characters (the letters A – D), you could specify in register DX for the set to start with the letter A, or the letter D. ES:BP is a pointer to the character set that you created.

When the value in register AL is 1, a normal monochrome character set (with characters defined in an 8 × 14 area) is loaded in memory. The value (0-3) in register BL determines the placement of the character set (set 1, set 2, etc.) in memory. When the value in register AL is 2, a standard character set (with characters defined in an 8 × 8 area) is loaded in memory. The value in register BL determines the placement of the character set in memory.

The values 10, 11, and 12 for register AL provide similar functions as for the values 0, 1, and 2 for register AL. The major differences are (for values 10, 11, and 12) that the CRT controller will be automatically reprogrammed to accommodate the operation, and that video page 0 must be currently displayed.

When the value placed in register AL is 20, you may define your own graphics character set as follows: the extended graphics card will assume that the number of scan lines per character will be in an 8 × 8 format, and will display the number of lines on the screen relative to what they were previously.

When the value placed in register AL is 21, it allows you to define your own graphics character set. Note that you must specify the entire character set, rather than just portions of it. The value placed in register CX indicates the number of scan lines per character (in an 8 × y format) that will occur, and the value placed in register BL indicates how many lines will be displayed on the screen as follows: when the value in register BL is 0, the number of lines on the screen will be relative to the value placed in register DL. When the value placed in register BL is 1, 14 lines will be displayed on the screen. When the value placed in register BL is 2, 25 lines will be displayed on the screen. When the value placed in register BL is 3, 43 lines will be displayed on the screen. ES:BP is a pointer to the custom graphics character set.

When the value placed in register AL is 22, it causes characters to be displayed in an 8 × 14 format; and when the value in AL is 23, it causes characters to be displayed in an 8 × 8 format.

When the value placed in register AL is 30, a request for one of six current configuration parameters can be made. Information pertaining to a specific parameter is returned in register ES:BP. The six parameters that can be checked relate to the value placed in register BH as follows: when the value placed in BH is 0, it returns a pointer to an address containing the first 128 characters (0 – 127) used in graphics modes. When the value in BH is 1, it points to the second 128 characters (128 – 256) used in graphics modes.

When the value in register BH is 2, it returns a pointer to an address that contains a character set with an 8 × 14 format. When the value in BH is 3, it returns a pointer to an address that contains a standard 8 × 8 (double-dot) character set for the first 128 characters (0 – 127). When the value in register BH is 4, it returns a pointer to an address that contains a standard 8 × 8 (double-dot) character set for the second 128 characters (128 – 256).

When the value in register BH is 5, it returns a pointer to a modification table that contains characters in a 9 × 14 format. Note that these characters are only those that differ from the characters in the 8 × 14 set. Most of these characters are identical.

Function Code 12H: Alternate Select — When the value placed in register BL is 10, this function code reports status information regarding the color video and monochrome modes, and the amount of memory present on the extended graphics card. This function can conveniently be used to poll the presence of the extended graphics card. If certain values are received, it can be assumed that the card produced them and is present

When the value in register BL is 20, an alternate print screen routine is enabled. This routine allows recognition of up to 43 lines of text (as provided by the extended graphics card in certain modes).

Function Code 13H: Write String — This function allows an entire string of characters to be written to the video display using a single MFM-200 function request. Several methods of writing a string are available. For any given method, registers ES and BP must indicate the "pointer" to the string to be written; register CX must indicate the length of the string to be written; DX must define the cursor position at which the string must begin; and BH indicates the page number that the string is to be written to. Register AL indicates various ways to write a string to the video display. When the value in register AL is 0 or 1, the contents of register BL indicates that the written string will consist of attributes. When the value in AL is 0, The string generated will be a list of ASCII characters, with the cursor left in its previous position. When the value of AL is 1, the string will be a list of ASCII characters with the cursor moved.

When the value in register AL is 2, the string generated will be a list of character and attribute pairs, with the cursor left in its previous position. When the value in register AL is 3, the string will be a list of character and attribute pairs, with the cursor moved.

Function Code 100H: Set Scroll Mode — This function code will cause the interrupt to select one of three scrolling modes. Load register AL with a value that represents the specified scrolling mode. Mode 0 is software scrolling, mode 1 is hardware "jump" scrolling, and mode 2 is hardware "smooth" scrolling. Note the following limitations.

- Hardware scrolling is not permitted while in 40 × 25 text mode.
- Hardware smooth scrolling may only be used in the high-resolution graphics modes.

- Hardware jump scrolling will only work in the graphics and 80 by 25 modes, including the monochrome mode.

- Software scrolling will work in all modes. If you run software that bypass the ROM calls, you should use software scrolling.

Note: The extended graphics card does not currently support hardware or smooth scrolling.

Function Code 101H: Set Interlace Mode — When the value in register AL is 0, a 25-line display providing characters formed in an 8 × 8 area is generated. When the value in register AL is 1, a 25-line display providing characters formed in an 8 × 16 area is generated. When the value in register AL is 2, the full interlace mode is implemented. This mode provides 50 lines of display but could possibly cause the display to flicker because the screen refresh rate is reduced to 30 cycles per second.

Note that the extended graphics card provides 43 lines of display (non-interlaced) that does not flicker. A high-resolution monitor is required in this mode of operation.

Function Code 102H: Compatibility Mode — This function code provides slower (IBM-compatible) video to accommodate applications programs that require it. When the value in register AL is FFH, the compatibility mode is implemented. When the value in register AL is 0, the video is returned to a faster operating mode.

Video Initialization (INT 1DH)

The INT 1DH instruction, unless otherwise programmed by an application program, will initialize the video section parameters according to the information stored in the BIOS ROM. This is the same information that is used to initialize the video section of the computer when the system is turned on.

Four tables are required to fully utilize this interrupt: modes 0 and 1, 40 × 25 text mode; modes 2 and 3, 80 × 25 text mode; modes 4, 5, and 6, graphics mode; and mode 7, monochrome video card mode. The default values for each CRTC register is provided in Table 11-4.

Table 11-4: Video Initialization Default Values

REGISTER NUMBER	HEXADECIMAL VALUES			
	40 × 25 TEXT	80 × 25 TEXT	GRAPHICS	MONOCHROME
R0	38	71	38	61
R1	28	50	28	50
R2	2D	5A	2D	52
R3	0A	0A	0A	0F
R4	1F	1F	7F	19
R5	06	06	06	06
R6	19	19	64	19
R7	1C	1C	70	19
R8	02	02	02	02
R9	07	07	01	0D
R10	06	06	06	0B
R11	07	07	07	0C
R12	00	00	00	xx
R13	00	00	00	xx
R14 – R17	xx	xx	xx	xx

xx — This value may be between 00H and FFH.

Graphics Characters (INT 1FH)

The INT 1FH instruction will allow you to access an extended character set for use with the medium- and high-resolution color graphics modes. Normally the first 128 (codes 00H – 7FH) used in the graphics modes are called from the ROM. In addition to these standard characters, you may create a custom character set of 128 (codes 80H – FFH) characters by using the following procedure.

1. Allocate a 1K section of memory (not video memory) to hold your characters set. You will need eight bytes for each character you define.

2. Define each character in an 8 × 8 matrix as shown in Figure 11-1. Note that the top line in the matrix buts up against the bottom line of the character above it. If you define any characters, allow for ascenders and descenders. Also allow for a space between characters. The illustration uses the number 7 as an example.

3. For each line of the matrix, identify which of the 8 pixels will be lit.

4. With the first pixel as the most-significant bit, add the binary weights of the lit pixels in each row to produce a decimal value for the byte representing that row.

5. Load all 8 bytes that form the defined character into the first 8 bytes of the memory allocated for the character set.

6. Repeat this procedure for each of the 128 characters in the set.

7. Once the characters have been defined and placed into memory, set the pointer at interrupt 1FH (memory location 0000:007C) to the start of memory allocated for the defined character set.

8. Execute an INT 1FH instruction. Now, whenever you call a character code between 80 and FF in graphics mode, the corresponding character from the set you created will be used in place of the firmware character normally used.

	BINARY VALUE								RESULTING
ROW	80H	40H	20H	10H	08H	04H	02H	01H	VALUE
1			•	•	•	•	•		3EH
2							•		02H
3							•		02H
4						•			04H
5					•				08H
6				•					10H
7				•					10H
8									00H

Figure 11-1: Character Design Matrix

Part IV

System Hardware

The chapters in this part of the manual provide an in-depth system hardware analysis of the Z-200 PC Series Computers. The electrical characteristics of each major assembly are analyzed. The text is supported with device pinouts, block diagrams, and schematics where applicable.

Note that this part of the manual covers only those assemblies provided in the basic computer. Refer to the Appendices for information pertaining to other hardware assemblies.

Chapter 12

Power Supply and Backplane

Power Supply

This section of the chapter provides information on the power supply used in the basic Z-248 computer. The power supply is **not serviceable**.

Specifications

Input Voltage:	115 VAC (100 VAC minimum to 130 VAC maximum). or 220 VAC (200 VAC minimum to 230 VAC maximum).
Input Line Frequency:	50 Hz or 60 Hz (± 3 Hz).
Input Line Current:	4 amp maximum continuous (full load).
Output Power:	200 watts maximum.

WARNING: Do not attempt to service the power supply. Serious or fatal injury may result.

A simplified block diagram of the power supply is illustrated in Figure 12-1. Refer to this block diagram during the following discussions in this section of the Chapter.

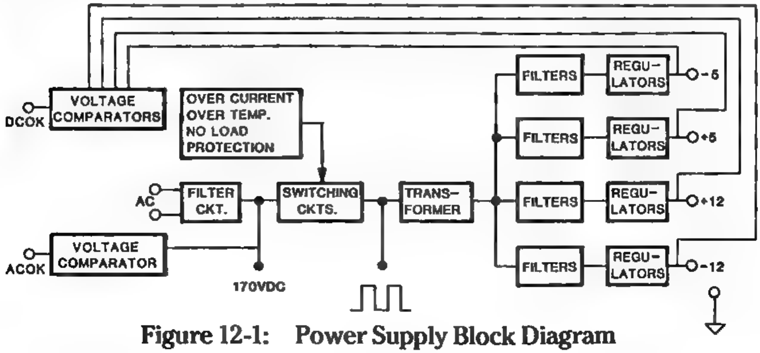


Figure 12-1: Power Supply Block Diagram

Line Potential

The power supply can operate on either 115 or 230 VAC, by a switch on the back panel of the computer. Refer to the following for additional information.

- Line select switch — Selects either 115 or 230 VAC operation. The switch is clearly marked regarding the potential selected. If the computer is used in the United States, the switch must be set for 115 VAC operation. If the computer is used in an area serviced by 230 VAC, the switch must be changed accordingly. Note that 230 VAC operation requires a different line cord. Make certain that the cord is the proper type and is rated for 230 VAC operation. Referenced from the front of the computer, this switch is located at the right, rear portion of the chassis, just above the AC receptacle and power switch.

- 115 VAC operation — When the line select switch is in the 115 VAC position, AC current is directed through the primary winding of an input transformer with a 1:1 turns ratio. 115 volts RMS (root of the mean of the square) will therefore be realized across the secondary of the transformer. This potential is conventionally rectified and filtered to derive approximately 170 volts DC. The high DC potential acts as input voltage to a high efficiency switching mode circuit.

- 230 VAC operation — When the line select switch is changed to the 230 VAC position, a second winding is connected in series with the primary winding of the input transformer. This second winding creates a 2:1 turns ratio. With AC current now directed through both primary windings, the secondary winding still develops only 115 volts RMS.

The input transformer has a high coefficient of coupling but is not a true power transformer. Its size and weight are therefore minimal. The switching circuits that invert the 170-volt DC potential create a quasi-square wave signal. The signal is applied to a second transformer (toroid) that steps the quasi-square wave down in voltage. The toroid creates the current delivering capabilities inherent in the entire supply (4 amps @ 115 VAC and 2 amps @ 230 VAC).

- On/Off power switch — This switch applies line voltage (115 VAC or 230 VAC) to the power supply unit. The line voltage is conventionally rectified and filtered to approximately 170 VDC. Switching circuits then process this potential into required operating voltages.

Outputs

Refer to Figure 12-2 for the wiring diagram of the power supply output voltage connectors.

All outputs from the precision-regulated power supply are electrically isolated from the main voltage and share one common ground, that is the chassis of the computer.

The operating voltage for floppy drives is interfaced through connectors P3 and P5. The operating voltage for Winchester drives is interfaced through connectors P2 and P4. The voltage used to power the disk drives have special requirements for output ripple frequency components in the frequency range of 60 to 120 kHz. To obtain the required low levels, a low pass filter is added to the regulator output leads. The output ripple voltage is measured with an oscilloscope with a bandwidth of 3 kHz. Component frequencies of the output ripple of the 5-volt or 12-volt outputs should not exceed 2.6 mV peak between the frequencies of 60 and 120 kHz.

Connector P1 interfaces power requirements, ground, and the ACOK and DCOK status signals to the backplane board. Refer to Table 12-1 for pinout and signal identification of connector P1, and to Figure 12-2 for the correct pin orientation.

Table 12-1: Connector P1 Pinout

PIN NUMBER	SIGNAL NAME
1,2	+12 VDC
3, 5, 6, 9, 12, 15	GND
4	−12 VDC
7	ACOK
8, 11, 14	+5 VDC
10	DCOK
13	−5 VDC

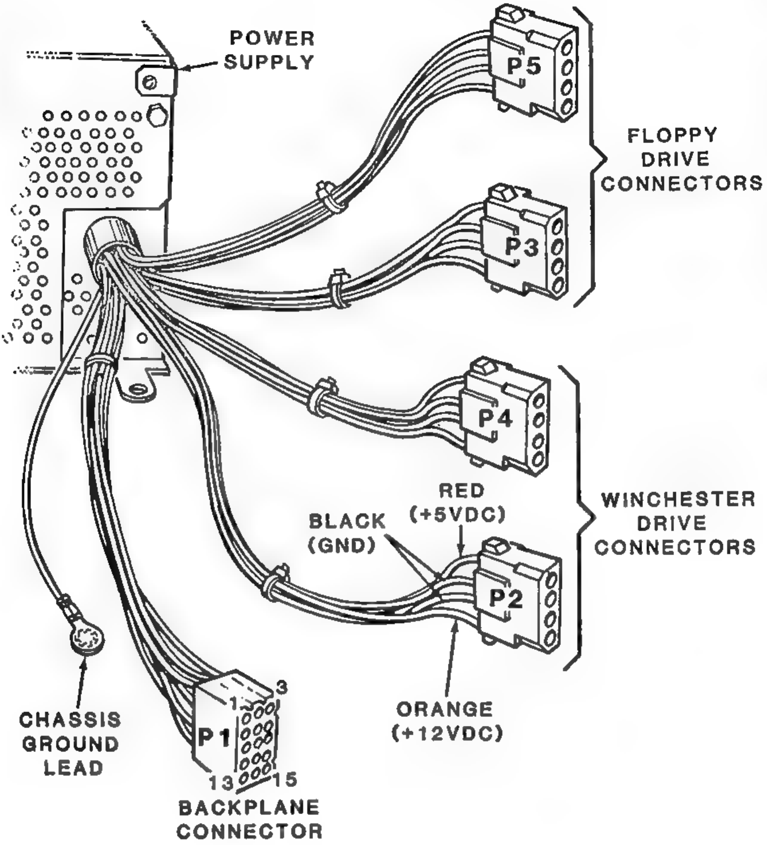


Figure 12-2: Power Supply Output Connectors

Backplane

This section provides information on the backplane used in the basic Z-248 computer. In general, the backplane interfaces cards to the system busses, and power to the cards from the power supply.

Specifications

Bus Interface: 16-bit PC AT compatible bus; 8-bit PC-compatible bus; and a special Zenith Data Systems bus.

Card Interface: Four PC AT compatible connectors; two PC-compatible connectors; four Zenith Data Systems connectors.

Keyboard Interface: 5-pin DIN connector mounted directly on the backplane board.

Battery Backup: 3.6-volt lithium battery to power the computer's real-time clock chip.

Refer to Figure A-1 in Appendix for the signal interface schematic (basic Z-248) of the computer backplane circuit.

The backplane provides the physical and electrical interface among cards used in the computer system. This backplane also interfaces power requirements to logic present on individual cards. A 16-bit address bus, 16-bit data bus, and control bus transfer addresses, data, and control signals among circuit cards.

The backplane accepts IBM PC AT compatible cards. Four slots that house the appropriate edge connectors are supplied for this purpose. In addition, two slots are equipped with IBM PC-compatible connectors. This allows for a high degree of special performance flexibility. Because Zenith Data Systems cards employ some special-purpose signals throughout the system, the remaining four slots on the backplane are equipped with two 62-pin connectors in each slot. Refer to Figure 12-3.

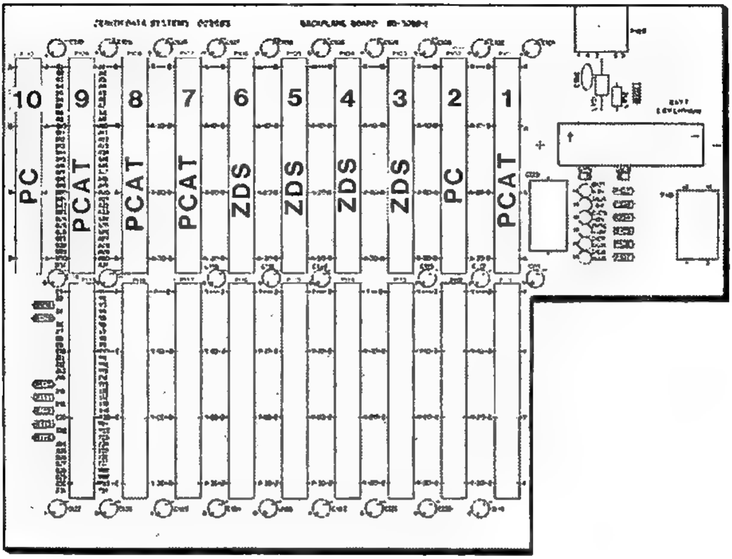


Figure 12-3: Backplane Slots

A battery backup circuit also is present on the backplane. When the computer is powered down, +5 VDC is derived from the battery. This voltage allows a special CMOS device to retain the status of data present within its registers. The information preserved is the system configuration data.

WARNING: Do not under any circumstances short-circuit or attempt to charge the lithium battery. Also do not force it to over-discharge. Do not disassemble, crush, penetrate, incinerate, or expose the battery to temperature above 93 degrees Celsius (200 degrees Fahrenheit). Otherwise, the lithium battery may leak and/or explode. If a battery is found to be leaking, do not get the liquid on your skin, as a severe chemical burn may occur.

The battery may swell and leak if operated at both maximum rated current and temperature.

Also mounted on the backplane are six diagnostic LEDs that monitor the status of the power supply voltage. The diagnostic LEDs are located in front of the 5-volt lithium battery and to the right of capacitor C129. The LED status indicators are positioned from the front to the rear of the computer as follows:

- D108 — ACOK (AC voltage present) status LED.
- D107 — DCOK (DC voltage within specifications) status LED.
- D106 — +12 VDC status LED.
- D105 — −12 VDC status LED.
- D104 — −5 VDC status LED.
- D103 — +5 VDC status LED.

NOTE: If a serious problem exists in the power supply, the ACOK (D108) or DCOK (D107) LEDs may not light at normal powerup. In addition, the diagnostic LEDs on the I/O card will all remain lit.

Bus Description

The Z-248 computer backplane contains a PC-compatible bus, PC AT compatible bus, and a Zenith Data Systems (ZDS) bus that carries special signals throughout the computer system. The busses are integrated into three different types of edge connectors (PC-compatible, PC AT compatible, and ZDS connectors). Figure 12-3 illustrates the arrangement of the connectors on the backplane. Table 12-2 details the pinout and description of the signals present on the PC-compatible connectors.

Table 12-2: PC-Compatible Connector Pinout

PIN	SIGNAL	
NUMBER	NAME	DESCRIPTION
A1	$\overline{\text{I/O CHCK}}$	Input/output channel check. Provides CPU with parity error status for memory or other input/output devices. Active-low indicates errors.
A2	SD7	System data bit 7.
A3	SD6	System data bit 6.
A4	SD5	System data bit 5.
A5	SD4	System data bit 4.
A6	SD3	System data bit 3.
A7	SD2	System data bit 2.
A8	SD1	System data bit 1.
A9	SD0	System data bit 0.
A10	I/O CHRDY	Input/output channel ready. Used by slower input/output devices to make sure that data is not lost during read and write operations. May be held low (not ready) up to 10 clock cycles (210 ns).
A11	AEN	Address enable. Assigns control of read and write operations to DMA controllers.
A12	SA19	System address bit 19.
A13	SA18	System address bit 18.
A14	SA17	System address bit 17.
A15	SA16	System address bit 16.
A16	SA15	System address bit 15.
A17	SA14	System address bit 14.
A18	SA13	System address bit 13.
A19	SA12	System address bit 12.
A20	SA11	System address bit 11.
A21	SA10	System address bit 10.
A22	SA9	System address bit 9.
A23	SA8	System address bit 8.
A24	SA7	System address bit 7.
A25	SA6	System address bit 6.
A26	SA5	System address bit 5.
A27	SA4	System address bit 4.
A28	SA3	System address bit 3.
A29	SA2	System address bit 2.
A30	SA1	System address bit 1.
A31	SA0	System address bit 0.
B1	GND	Ground.

Table 12-2 (continued): PC-Compatible Connector Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
B2	$\overline{\text{RESET}}$	When low, initializes the system logic.
B3	+5 VDC	+5 VDC bus.
B4	IRQ9	Interrupt request 9. (IRQ2 in PC-compatible computers. Not used, but available for assignment to a user-selected device.)
B5	−5 VDC	−5 VDC bus.
B6	DRQ2	DMA request 2. Assigned to floppy disk controller.
B7	−12 VDC	−12 VDC bus.
B8	0WS	This signal is generated to inform the CPU of a fast memory access using zero wait state technology.
B9	+12 VDC	+12 VDC bus.
B10	GND	Ground.
B11	$\overline{\text{SMEMW}}$	System memory write. When low, causes data on the data bus to be stored in memory.
B12	$\overline{\text{SMEMR}}$	System memory read. When low, causes memory to drive data onto the data bus.
B13	$\overline{\text{IOW}}$	Input/output write. When low, instructs an input/output device to read data on the data bus.
B14	$\overline{\text{IOR}}$	Input/output read. When low, instructs input/output device to drive its data onto the data bus.
B15	$\overline{\text{DACK3}}$	DMA acknowledge 3. Assigned to Winchester drive controller.
B16	DRQ3	DMA request 3. Assigned to Winchester drive controller.
B17	$\overline{\text{DACK1}}$	DMA acknowledge 1. Not used, available for user assignment.
B18	DRQ1	DMA request 1. Available to the user.
B19	$\overline{\text{REF}}$	System refresh enable. Initiates memory refresh.
B20	SCLK	6 MHz system clock.
B21	IRQ7	Interrupt request 7. Assigned to parallel interface.
B22	IRQ6	Interrupt request 6. Assigned to floppy controller.

Table 12-2 (continued): PC-Compatible Connector Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
B23	IRQ5	Interrupt request 5. Assigned to Winchester drive controller.
B24	IRQ4	Interrupt request 4. Assigned to serial port (configurable).
B25	IRQ3	Interrupt request 3. Assigned to serial port (configurable).
B26	$\overline{\text{DACK2}}$	DMA acknowledge 2. Assigned to floppy controller.
B27	T/C	Terminal count. Goes high when terminal count for any DMA channel is reached.
B28	ALE	Address latch enable. Generated by the bus controller to indicate valid processor addresses to the input/output channel.
B29	+5 VDC	+5 VDC bus.
B30	OSC	A 14.31818 MHz oscillator provides the basic timing for the system.
B31	GND	Ground.

The PC AT compatible connectors incorporate the signals detailed in Table 12-2 (A1 through A31, and B1 through B31) and the signals detailed in Table 12-3 (Y1 through Y18, and Z1 through Z18).

Table 12-3: PC AT Compatible Connector Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
Y1	$\overline{\text{SBHE}}$	System bus high enable. This signal indicates data transfer over the D8 – D15 bits of the data bus.
Y2	SA23	System address bit 23.
Y3	SA22	System address bit 22.
Y4	SA21	System address bit 21.
Y5	SA20	System address bit 20.
Y6	SA19	System address bit 19.
Y7	SA18	System address bit 18.
Y8	SA17	System address bit 17.
Y9	$\overline{\text{MEMR}}$	Memory read. When low, allows memory to place data onto the data bus.

Table 12-3 (continued): PC AT Compatible Connector Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
Y10	$\overline{\text{MEMW}}$	Memory write. When low, allows data on the bus to be stored in memory.
Y11	SD8	System data bit 8.
Y12	SD9	System data bit 9.
Y13	SD10	System data bit 10.
Y14	SD11	System data bit 11.
Y15	SD12	System data bit 12.
Y16	SD13	System data bit 13.
Y17	SD14	System data bit 14.
Y18	SD15	System data bit 15.
Z1	$\overline{\text{M16}}$	16-bit memory cycle. When this signal is active, it allows a 16-bit, 1 wait state memory cycle.
Z2	$\overline{\text{I16}}$	16-bit I/O cycle. When this signal is active, it allows a 16-bit, 1 wait state I/O cycle.
Z3	IRQ10	Hardware interrupt request 10.
Z4	IRQ11	Hardware interrupt request 11.
Z5	IRQ12	Hardware interrupt request 12.
Z6	IRQ13	Hardware interrupt request 13.
Z7	IRQ14	Hardware interrupt request 14.
Z8	$\overline{\text{DACK0}}$	DMA acknowledge 0. Goes active when the CPU acknowledges the corresponding DMA cycle request signal (DREQ0).
Z9	DREQ0	DMA request 0. This signal is used by a peripheral device that requires DMA service.
Z10	$\overline{\text{DACK5}}$	DMA acknowledge 5. Goes active when the CPU acknowledges the corresponding DMA cycle request signal (DREQ5).
Z11	DREQ5	DMA request 5. This signal is used by a peripheral device that requires DMA service.
Z12	$\overline{\text{DACK6}}$	DMA acknowledge 6. Goes active when the CPU acknowledges the corresponding DMA cycle request signal (DREQ6).
Z13	DREQ6	DMA request 6. This signal is used by a peripheral device that requires DMA service.
Z14	$\overline{\text{DACK7}}$	DMA acknowledge 7. Goes active when the CPU acknowledges the corresponding DMA cycle request signal (DREQ7).

Z15	DREQ7	DMA request 7. This signal is used by a peripheral device that requires DMA service.
Z16	+5V	+5VDC bus.
Z17	$\overline{\text{MASTER}}$	This signal is used in conjunction with a DMA request signal to allow a processor or DMA controller to take control of the system busses.
Z18	GND	Ground.

The Zenith connectors incorporate the signals detailed in Table 12-2 (A1 through A31, and B1 through B31), the signals detailed in Table 12-3 (Y1 through Y18, and Z1 through Z18), and the signals detailed in Table 12-4 (Y19 through Y31, and Z19 through Z31).

Table 12-4: Zenith Data Systems Connector Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
Y19	DCOK	DC voltage within tolerance. This signal causes the appropriate diagnostic LED on the backplane to remain on during normal operation. If this LED is off, the DC voltage produced by the power supply is out of tolerance.
Y20	B5V	Backup battery voltage. When the computer is operating, +5 VDC is applied to this line to power the real-time clock chip. When the computer is turned off, +3.6 VDC is applied to this line (by the lithium battery) to power the real-time clock chip.
Y21	ACOK	AC voltage present. This signal causes the appropriate LED on the backplane to remain on as long as AC voltage to the power supply is present.

Table 12-4 (continued): Zenith Data Systems Connector Pinout

PIN	SIGNAL	
NUMBER	NAME	DESCRIPTION
Y22	Z100EN	Z-100 mode graphics enable. This function is not currently supported in the Z-241 computer.
Y23	KBDATA	Keyboard data. Data transmission from the keyboard is interfaced to the I/O card on this line. This signal is synchronous with the keyboard clock.
Y24	KBCLOCK	Keyboard clock.
Y25	ERROR	Error condition. This signal is generated whenever an error condition is encountered in the computer.
Y26	CLRERR	Clear error. This signal causes the ERROR signal to be reset (inactive).
Y27	$\overline{IOCLK}$	I/O clock. This 6 MHz signal is derived from the system clock signal.
Y28	$\overline{S2F}$	Memory or I/O select. The status of this signal distinguishes memory access from I/O access. This signal (in conjunction with other signals) can be used to place the CPU in a halt mode.
Y29	FREE	System busses available. When the free signal is active, it indicates that the system busses are available for use by a DMA controller or I/O channel processor.
Y30	$\overline{DEN}$	Data enable. This signal enables transceivers on the data bus so that data transfers can occur.
Y31	$\overline{INTA}$	Interrupt acknowledge. This signal is used to distinguish interrupt acknowledge cycles from I/O cycles.
Z19	$\overline{WRT}$	Write. This signal is asserted during a memory write cycle.
Z20	$\overline{MEM}$	Memory. This signal is asserted during read, write, and refresh operations involving memory.
Z21	$\overline{FAST}$	Fast memory access. The fast memory access signal is generated to inform the CPU that the operation to be performed will occur without wait states.

Table 12-4 (continued): **Zenith Data Systems Connector Pinout**

PIN NUMBER	SIGNAL NAME	DESCRIPTION
Z22	SLOW	Slow memory access. The SLOW signal allows wait states to be inserted into memory accesses to allow applications programs that may be time dependent to operate correctly.
Z23	NMI	Non-maskable interrupt request. This signal causes CPU program execution to be driven to a type 2 vector address.
Z24	$\overline{\text{RC}}$	Clear DCOK. Whenever a software reset occurs, this signal will cause the DCOK (DC voltage within tolerance) signal to be reset.
Z25	$\overline{\text{DISPG}}$	Disable parity. This line is used to disable parity generation and checking during diagnostic tests in the Z-248 computer.
Z26	$\overline{\text{DEC7}}$	Decode address bits 5, 6, and 7. This signal is gated with various other signals to derive the numeric processor extension select signal (NPS1).
Z27	IRQ13	Interrupt request 13. When asserted, this interrupt request line indicates an error condition in the operation that the numeric processor extension is attempting to perform.
Z28	$\overline{\text{S1F}}$	80286 processor status signal. Used during various bus cycles.
Z29	PCLK	Processor Clock. This 8 MHz signal is used for timing throughout the computer system.
Z30	GATEA20	Gate address bit A20. Whenever addressing above the 1 megabyte boundary is to occur, this signal is asserted to cause the addressing to wrap to the beginning of that boundary. This is done because many applications software packages (PC-compatible) rely on this wrap function.
Z31	INTR	Maskable hardware interrupt request. This signal causes the CPU to suspend its current program execution and service a request being made by some device.

Chapter 13
CPU/RAM Card

This chapter describes the major circuits of the CPU/RAM (central processing unit/random access memory) card. A simplified block diagram of the CPU/RAM card is illustrated in Figure 13-1. Refer to this block diagram during the following discussions in this section of Chapter 13.

Specifications

CPU:	True 16-bit architecture
Microprocessor:	68-pin 80286
Clock Frequency:	8 MHz
Base Memory (RAM):	512K, standard. Expandable to 640K.

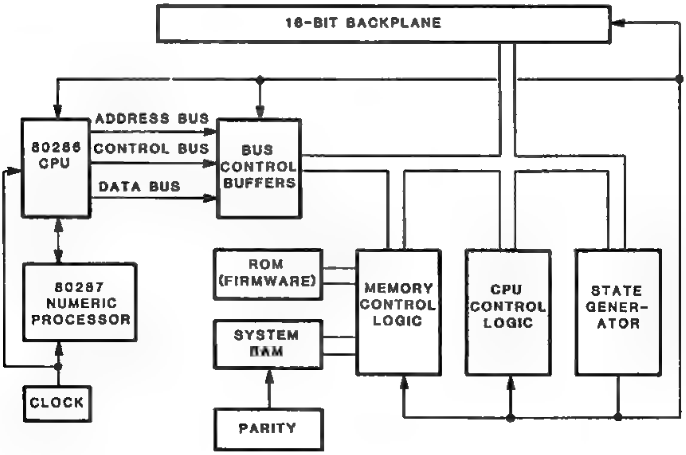


Figure 13-1: CPU/RAM Card Block Diagram

Central Processing Unit

The CPU portion of this card uses the 8 MHz 16-bit 80286 microprocessor. This high-performance VLSI (very-large-scale integration) device is designed with a 16-bit data bus and has addressing capabilities of up to 16 megabytes of memory. An extensive instruction set is available for the 80286 that incorporates a high level of flexibility and system performance.

Fundamentally, the 80286 is a 16-bit device but can also operate within an 8-bit architecture application. The 80286 exercises all of the functions of a typical 8-bit microprocessor (8088) at an increased rate.

The instruction set defines what particular operation the CPU is to perform and, as mentioned, is extremely flexible. Instructions may be stored permanently in ROM (firmware), or may be transferred from disk to the system RAM. In either case, instructions must be provided for the 80286 or it will not function.

Address Bus

Refer to Figure 13-2 for the schematic of the address bus circuit.

This bus is 24 bits wide (A0 – A23) and is used to locate a specific device or memory location. The microprocessor is capable of operating within two modes, each of which allows for different addressing capability.

- Real address mode — In this mode, the 80286 can address up to 1 megabyte of memory over a 20-bit address bus.

- Protected address mode — In this mode, up to 15 megabytes of memory may be addressed over a 24-bit address bus.

Microprocessor U218 controls all of the activities within the computer system. The CPU interfaces with portions of the computer by addressing and enabling various devices over its address and control busses. When the CPU places an address on the CPU_ADDR bus, transparent latches U269, U270, and U272 transfer the address onto the ADDR_BUS after completion of the current CPU cycle. However the transparent latches do not latch the address immediately.

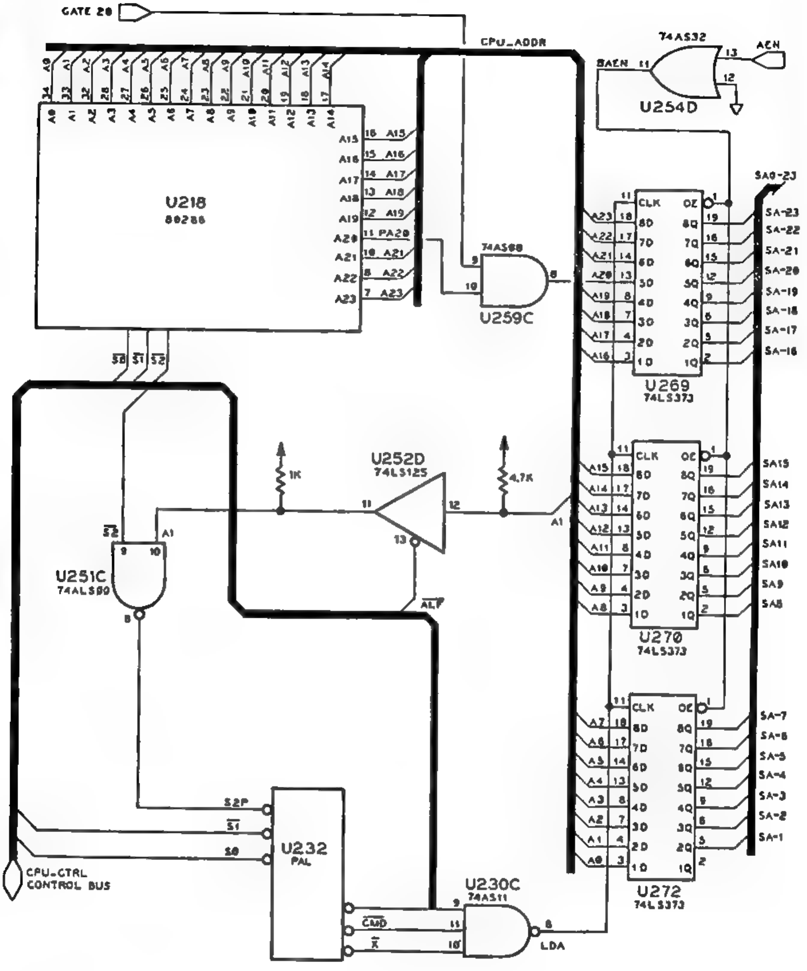


Figure 13-2: Address Bus Schematic

The address at the outputs of transparent latches U269, U270, and U272 is latched on the high-to-low transition of the LDA (load address) signal. The CPU asserts the $\overline{\text{S0}}$, $\overline{\text{S1}}$, and $\overline{\text{S2}}$ ($\overline{\text{S2}}$ selects memory or I/O) status signals onto the control bus. When the $\overline{\text{ALF}}$ signal is active low, the $\overline{\text{S2}}$ signal is gated with the A1 signal by pin 8 of U251 to produce the S2P signal. The $\overline{\text{S0}}$, $\overline{\text{S1}}$, and S2P signals then are transmitted to logic array U232. This logic array is programmed as a state machine that cycles the signals on its output lines in predetermined sequences. When the $\overline{\text{S0}}$, $\overline{\text{S1}}$, and S2P signals are all high, the state machine logic array (U232) is idle.

The state machine monitors the status signals and cycles through its various "states" determined by the operation being performed. In this case, the operation is generating an address onto the ADDR_BUS. When the "state" is produced that drives the $\overline{\text{ALF}}$, $\overline{\text{CMD}}$, or $\overline{\text{X}}$ signals true, the output of NAND gate U230 is driven low, producing the LDA (load address) signal. The active LDA signal acts as a clock input to the transparent latches residing on the CPU_ADDR bus and causes them to latch the status of the address at their outputs during its high-to-low transition. The latched address and control signals (S0F, S1F, S2F, BHEF, and CODF) then remain on the system ADDR_BUS and control bus until the end of the cycle.

Data Bus

Refer to Figure 13-3 for the schematic of the data bus circuit.

The CPU_DATA bus is 16 bits wide and interfaces directly to the 80287 numeric processor extension data bus. The D0 – D7 bits of the CPU_DATA bus are connected to an octal bidirectional latch (U239) whose output produces the D0 – D7 bits of the system DATA_BUS. The high order bits (D8 – D15) also are connected to a bidirectional latch (U240) and form the high order bits of the DATA_BUS. All of the data bits comprising the DATA_BUS are pulled-up by 4.7k ohm resistors inherent in resistor packs RP207 and RP208.

The $\overline{\text{S1F}}$ (data direction) signal transmitted over the control bus (CPU_CTRL) determines direction of data transfer through U239 and U240. The CPU and CPU_CTRL (CPU control) logic array (U232) control data transfers to and from either low order memory or high order memory by enabling or disabling either of data transceivers U239 or U240.

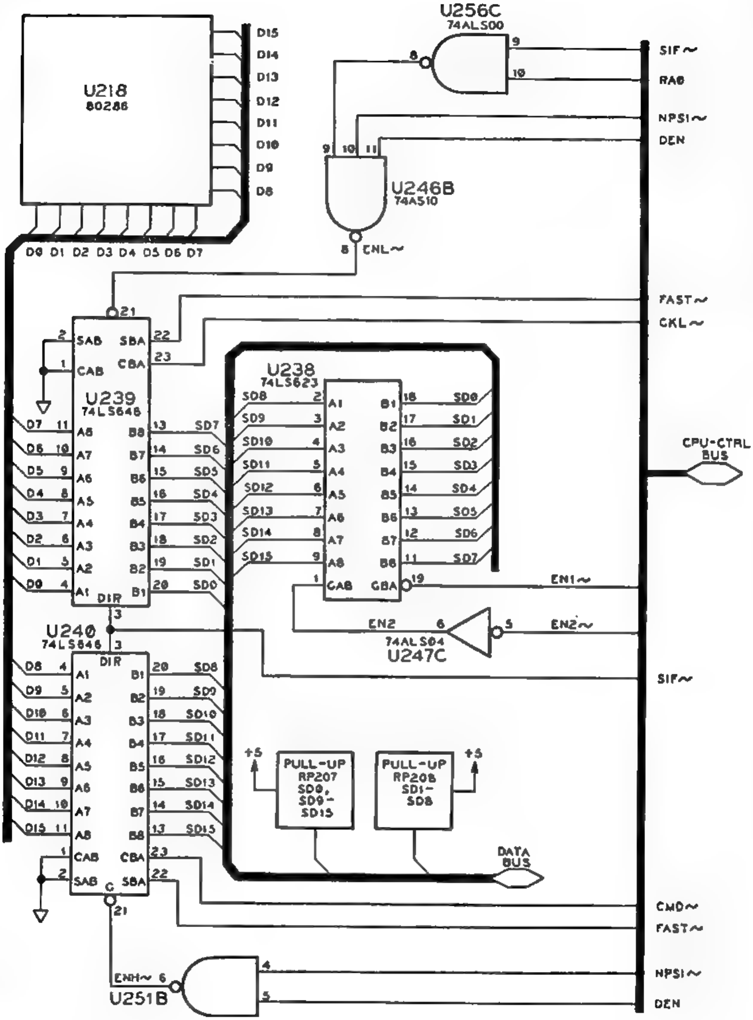


Figure 13-3: Data Bus Schematic

The RA0 signal present on the control bus is NANDed with the $\overline{\text{S1F}}$ signal on the control bus at pin 8 of U256. The output of this gate enables pin 9 of NAND gate U246. The DEN (data enable) signal and the $\overline{\text{NPS1}}$ signal, both present on the control bus, are then NANDed by pin 8 of U246 to produce the $\overline{\text{ENL}}$ (enable low byte data transceiver) signal. When true, the $\overline{\text{ENL}}$ signal enables data transceiver U239. The status of the $\overline{\text{S1F}}$ signal allows the CPU to read or write into low-order memory.

The DEN (data enable) signal is NANDed with the $\overline{\text{NPS1}}$ signal by pin 6 of U251 to produce the $\overline{\text{ENH}}$ (enable high byte data transceiver) signal. The $\overline{\text{ENH}}$ signal is applied to pin 21 of data transceiver U240 and when true, enables the device. The status of the $\overline{\text{S1F}}$ allows the CPU to read or write into high-order memory.

During a DMA transfer to or from I/O circuits, octal transceiver U238 is used to transfer information present on the low-order data bits of the DATA_BUS onto the high-order bits (D8 – D15). In this way, I/O DMA operations can occur as 16-bit functions rather than 8-bit functions.

The $\overline{\text{EN1}}$ signal is decoded by logic array U258 (Figure A-4 in Appendix A) to allow 8-bit I/O and memory read transfers to occur through U238. The $\overline{\text{EN1}}$ signal also allows 16-bit (two 8-bit bytes) DMA transfers from I/O devices to memory.

The $\overline{\text{EN2}}$ signal is also decoded by logic array U258 (Figure A-4 in Appendix A) to allow 8-bit write operations to occur from the CPU and memory to I/O devices. The $\overline{\text{EN2}}$ signal also allows 16-bit DMA transfers from memory to I/O devices.

16-bit words provided by the CPU can be broken into two 8-bit bytes that the DMA controllers and I/O devices can recognize. Subsequently, two 8-bit bytes provided by the DMA controllers and I/O devices may be redistributed as a 16- bit word so that the 80286 CPU can work faster.

Control Bus

Refer to Figure A-2 in Appendix A for the schematic of the control bus circuit.

The CPU_CTRL (control bus) used by the 80286 CPU carries various control and command signals to many different areas of the computer system. The CPU_CTRL bus directly interfaces the 80286 to the control inputs of the 80287 numeric processor extension. In addition, many other devices interfaced to this bus can use it to provide input to the CPU. Signals such as CLK286, $\overline{\text{RDY286}}$, and RST286 are present on the CPU_CTRL bus. Logic gates residing on the bus decode control and command signals used by other parts of the computer system.

The 80286 Instruction Cycle

The microprocessor implements instruction cycles through four internal independent units that operate simultaneously. Each unit has specific responsibilities but work they together to induce instruction fetches, data decoding, and instruction execution. This type of architecture is referred to as a "pipeline structure." The pipeline is divided into four areas as follows:

- Bus unit — This unit controls the various bus operations for the CPU. The bus unit contains individual circuits that latch and drive address bytes onto the address bus, and also contains data transceivers so that data can be read from and written to the main data bus. Various control logic is also contained within the bus unit to generate control signals to the system hardware.

  When the bus unit is not executing bus instructions, another part of the unit fetches the next six sequential instruction bytes from memory. The active logic assumes that the next instruction in the program follows the current instruction in memory. If a subroutine is introduced through software, the active logic branches to the specified memory location and continues to preload instructions found there.

  The instructions fetched are stored in the bus unit until they are required for use by the "instruction unit." By having the next six bytes of data readily available to the instruction unit, system throughput can be greatly advanced, as the CPU does not have to wait for each sequential instruction to be individually fetched from memory.

- Instruction unit — This unit accepts the stored instructions from the bus unit and decodes them. When fully decoded, the instructions are placed in a register so that they are available for use by the "execution unit."

- Execution unit — The execution unit uses decoded instructions from the instruction unit to execute data transfers to or from memory, registers, or I/O devices.

- Address unit — This unit provides memory management for the CPU and also supplies the bus unit with address bytes that it drives onto the address bus.

There are two basic functions during an instruction cycle, the instruction fetch and the execution of the instruction. Instruction fetches are identical but the execution of the instruction may be of variable length depending on the instruction. Using the pipeline architecture allows greater system throughput as illustrated in Figure 13-4.

During an instruction fetch, the contents of the program counter indicates to the CPU which memory location to address to read an instruction. Each time an instruction fetch occurs, the program counter increments so that it points to the memory location of the next instruction.

NOTE: While reading the following section you may need to refer to the CPU control logic schematic (parts 1 and 2) illustrated in Figure A-3 and Figure A-4 respectively, of Appendix A.

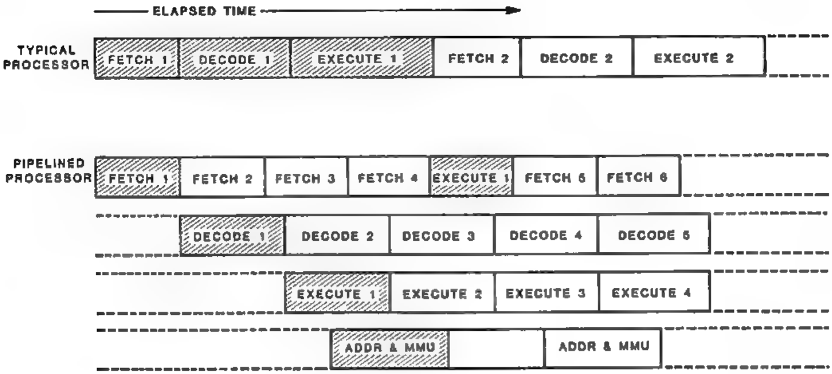


Figure 13-4: Sequential vs Pipeline Processors

The CPU addresses a memory location determined by the program counter, by placing address bits and control signals onto the CPU_ADDR (CPU address) and CPU_CTRL (CPU control) busses. The status of the address bits are immediately conducted onto the ADDR_BUS (main system address bus) through transparent latches. Although the address is present on the bus, it is not active until the CPU status signals ($\overline{S0}$, $\overline{S1}$, and S2) instruct logic array U232 to decode the true states of the $\overline{ALF}$, $\overline{CMD}$, and $\overline{X}$ control signals. These three signals are ANDed by pin 8 of U230 and driven onto the control bus as the LDA (load address) signal. During the high-to-low transition of the LDA signal, the transparent latches retain the status of the bits present at their outputs.

When the AEN (address enable) signal is low, the address on the bus selects a particular memory or I/O location. If the AEN signal is high, it indicates that a DMA data transfer or refresh cycle is occurring. The high status of AEN disables the transparent latches so that bus contention does not occur.

When memory logic detects a valid address, the appropriate control signals are asserted that allows the addressed memory location to be read. The instruction read from memory is placed in the bus unit of the 80286 via the data bus. The instruction is then transferred from the bus unit to the CPU through internal data transceivers.

I/O (input/output) operations are identical to memory operations except for the state of the M/$\overline{IO}$ (memory/IO) signal. When this signal is high, memory will be addressed and when it is low, I/O circuits will be addressed. Table 13-1 indicates control logic status and the bus cycle operation induced by the control logic.

Table 13-1: Bus Cycle Control Logic

BUS CYCLE	$\overline{S0}$	$\overline{S1}$	M/$\overline{IO}$	COD/$\overline{INTA}$
Interrupt acknowledge	L	L	L	L
Memory data read	H	L	H	L
I/O data read	H	L	L	H
Memory instruction read	H	L	H	H
Memory data write	L	H	H	L
I/O data write	L	H	L	H

Interrupt Requests

Two different types of interrupts can be used with the 80286 CPU, maskable (INTR) and non-maskable (NMI). The non-maskable interrupt cannot be ignored by the CPU, while some instructions can tell the CPU to ignore certain maskable interrupts. In all cases, the basic operation of all interrupts is the same.

Maskable

Pin 57 of the 80286 is the INTR (maskable hardware interrupt request) input to the CPU. This is a flexible type of interrupt because the interrupting device requests the CPU to execute the service routine. When the CPU acknowledges that this type of interrupt has occurred, the PIC (programmable interrupt controller) supplies the CPU with the address of the instruction that begins the programming routine for that interrupt. As many as 256 different interrupting devices can use this input so long as each device provides a different address to the CPU. Interrupt controllers U350 and U353 (I/O card) prioritize up to 15 interrupt requests. When more than one interrupt request occurs at the same time, the highest priority interrupt will be processed first.

Whenever the CPU receives an interrupt request, it finishes execution of its current instruction; then acknowledges the interrupt by driving its COD/$\overline{\text{INTA}}$ line low. During the next bus cycle, the interrupt controllers deliver a vector or "type" address to the CPU. The address provided by the interrupt controllers is used to access a memory location containing a subroutine instruction set. The CPU executes the instructions found there and then returns to the program where it left off.

Non-Maskable

If an NMI (non-maskable interrupt) request occurs, the CPU is automatically driven to the address corresponding to a "type 2" interrupt. The CPU does not generate an external interrupt acknowledgment signal. In all cases, the non-maskable interrupt has priority over the maskable interrupt. When both types of interrupts occur at the same time, the non-maskable interrupt will be processed first. If another NMI occurs during the time that the initial NMI is being processed, it will be processed after completion of the initial NMI.

NOTE: When the 80286 operates in the real address mode, all interrupt requests are automatically masked when an NMI is being serviced. In the protected mode, the subroutine is entered through a task gate. During the subroutine, instructions may be executed with the maskable interrupt line either enabled or disabled. At the end of the service, an interrupt return command must be used to return the CPU to its previous program.

Table 13-2 and Figure 13-5 define the signal names and device pinout of the 80286 microprocessor.

Table 13-2: 80286 Device Signals

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	$\overline{\text{BHE}}$	This output indicates transfer of data on bits D8 – D15. Byte-oriented devices residing on bits D8 – D15 use $\overline{\text{BHE}}$ to condition chip select functions.
4, 5	$\overline{\text{S1}}$, $\overline{\text{S0}}$	These outputs indicates initiation of a bus cycle and, along with M/$\overline{\text{IO}}$ and COD/$\overline{\text{INTA}}$, defines the type of bus cycle.
6, 61	$\overline{\text{PEREQ}}$, $\overline{\text{PEACK}}$	When $\overline{\text{PEREQ}}$ is high, the 80287 request the 80286 to allow it to assume program execution for a numeric operation. The 80286 asserts the $\overline{\text{PEACK}}$ signal low to acknowledge the request.
7,8, 10-28,	A23 – A0	24-bit address bus used to address physical memory and I/O ports. See Figure 13-5.
29-34	RESET	This input causes a system reset to clear the internal logic of the 80286 (active high).
30, 62	VCC	+ 5 VDC power supply.
31	CLK	6 MHz system clock. Provides fundamental timing for the operation of the 80286.
35, 60	VSS	System ground.
36-51	D0 – D15	System 8/16-bit data bus. See Figure 13-5.
52	CAP	A .047 uF capacitor must be connected between this pin and ground. The capacitor filters the output of the internal bias generator.
53	$\overline{\text{ERROR}}$	This error is relative to the numeric processor extension. When active, this signal causes the CPU to perform a processor extension interrupt during WAIT.

Table 13-2 (continued): 80286 Device Signals

54	$\overline{\text{BUSY}}$	This signal indicates the operating status of the numeric processor extension. An active $\overline{\text{BUSY}}$ signal causes the 80286 to stop its program execution and enter the WAIT mode.
57	INTR	This line conducts interrupt requests to the 80286 to suspend its current program execution and enter a subroutine program.
59	NMI	This line conducts non-maskable interrupt requests to the 80286 which is then forced to an interrupt vector type 2 address.
63	$\overline{\text{READY}}$	Bus cycles are extended without limit until the $\overline{\text{READY}}$ signal is asserted true. This will cause the bus cycle to terminate.
64, 65	HOLD, HOLDA	The bus hold request and hold acknowledge signals control bus activity.
66	COD/$\overline{\text{INTA}}$	Code/interrupt acknowledge distinguishes instruction fetch cycles from memory data read cycles. This output also distinguishes interrupt acknowledge cycles from I/O cycles.
67	M/$\overline{\text{IO}}$	Distinguishes memory access from I/O access.
68	$\overline{\text{LOCK}}$	When active, indicates that other system bus control devices are not to gain control of the system bus following the current bus cycle.

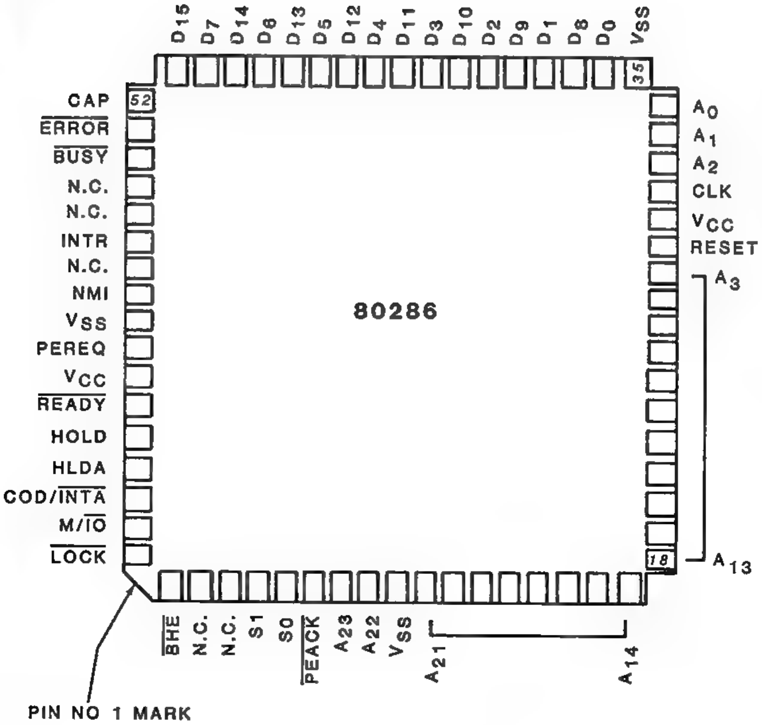


Figure 13-5: 80286 Device Pinout

CPU Control Logic

Refer to Figure A-3 and A-4 in Appendix A for the schematic of the CPU control logic circuit.

The CPU uses control logic to generate command signals and timing sequences required for proper access to different parts of the computer system. Some of the devices are buffers and latches that control the signals present on both the address and data busses.

The most important device responsible for generation of critical timing sequences is state machine logic array U232. This device generates predetermined states that enable or disable particular logic circuits. During the time duration of a given state, specific operations may be executed. This key element emulates the 80288 bus controller in that it controls all of the timing generations during program execution.

For example, when either the $\overline{\text{ALF}}$, $\overline{\text{CMD}}$, or $\overline{\text{X}}$ lines go low, the output of AND gate U230 is driven low, providing the LDA (load address) signal onto the CPU_CTRL bus. The LDA signal forces flip-flops residing on the CPU_ADDR bus to latch the address present on that bus to the ADDR_BUS. The state machine changes its state to condition certain logic circuits to allow memory, I/O, or DMA operations to occur. This state machine logic array always changes states to accommodate different modes of operation, but in some cases must remain in a given state until a particular operation is fully executed.

Logic array U258 decodes program execution signals. This logic array differs from the "state machine" in that it does not establish timing generations but merely decodes data. U258 decodes the interrupt acknowledge ($\overline{\text{INTA}}$) signal, as well as the $\overline{\text{EN1}}$ and $\overline{\text{EN2}}$ signals, and drives them onto the CPU_CTRL bus. Other devices in the computer that reside on the CPU_CTRL bus are then able to implement the signals, depending on which program execution is being performed. Various other decoded signals are gated by logic devices to produce specific required signals.

Devices reside on the 16-bit CPU_DATA bus to control proper timing and execution of operations, such as when a DMA operation between I/O and memory occurs. The bus must not be in contention with any other portion of the system during a data transfer, or data will be lost or overwritten, which in most cases will cause a system lock-up.

The 80286 CPU uses its CPU_CTRL bus to provide control devices with chip select and data direction signals to enable precise timing and gating of data. As previously mentioned, the logic array state machine is the key element of the timing network.

Fast Mode

This mode of operation allows the 80286 CPU to run without having wait states inserted into its memory accesses. The key element is the state machine that sets predetermined states in advance of CPU program execution to enable particular operations. The CPU can therefore run with fewer machine cycles and operate much faster.

NOTE: The fast mode of operation can only be enabled when specific modifications are made to discreet components on the CPU/RAM card and faster memory chips are used.

Slow Mode

The Z-248 computer is designed to operate so software that may be time-dependent will function properly under the 80286 CPU program execution. During this mode of operation, wait states are implemented internally to the state machine logic array. The presence of the wait states hold the $\overline{\text{RDY}}$ (ready) line inactive and the $\overline{\text{ALF}}$, $\overline{\text{CMD}}$, and $\overline{\text{X}}$ signals in their current state for a predetermined period of time. When this timeout is reached, the state machine is allowed to finish execution of its cycle. The wait states are used primarily for disk drive operations because additional time is required for time-dependent software to be executed properly.

To configure the computer for the slow mode, enter o64,B1 (OUT command to port 64H) at the monitor prompt, then boot the operating system. The computer will automatically insert the required wait states into its memory access cycles. This time delay is approximately equal to two 8088 memory access cycles.

To return the computer to the fast mode of operation enter o64,B2 at the monitor prompt, then boot the operating system.

NOTE: If the Z-248 computer is modified to run in the fast (zero wait state) mode of operation and problems are experienced with some software packages or hardware options, it may be necessary to run the computer in either the PC AT compatible mode (1 wait state), or the slow mode for the program to execute properly. Refer to Chapter 5, "Configuration" in Part II of this manual for more specific information.

80287 Numeric Processor Extension

The 80287 may be installed onto the CPU/RAM card as an option. The instruction set for the 80287 is an extension that contains over fifty additional instructions. The internal architecture of the numeric processor extension allows it to perform complex mathematical operations with fewer program instructions.

The 80286 can perform all of the operations of the 80287 but involves many more machine cycles for each operation, and therefore considerably more time. During some actual operations, the 80286 alone may not be able to execute the task. In all cases, the 80287 increases system throughput (assuming the software uses this device).

The 80287 numeric processor extension is connected directly to the status lines of the 80286 CPU through the CPU_CTRL bus. In this way, it can continually monitor operations performed by the CPU. When the 80286 encounters a need to execute a numeric instruction, the 80287 automatically recognizes the fact and initiates a PEREQ (processor extension request) signal to pin 61 of the 80286 CPU. The CPU completes its current instruction and then outputs the $\overline{\text{PEACK}}$ (processor extension acknowledge) signal. When the 80287 receives the acknowledgment, it drives its BUSY287 signal line low, and applies it to pin 11 of the CMNDCTRL logic array U253. This device generates the $\overline{\text{BUSY286}}$ signal that indicates to the CPU that the 80287 is beginning to exercise its internal instruction set to process the numeric operation.

Data transfers initiated by the 80287 are processed in a fashion similar to a DMA operation. However, the 80286 CPU has exclusive control over the 80287 during a data transfer.

If the 80286 CPU generates specific I/O addresses (00F8H, 00FAH, and 00FCH) during an instruction execution, the 80287 automatically responds by issuing another PEREQ (processor extension request) signal to the 80286. The 80286 then will check the $\overline{\text{BUSY}}$ line of the 80287. If it is still executing a previous routine, the CPU will wait until the numeric processor is free before it issues the acknowledge ($\overline{\text{PEACK}}$) signal. The 80287 is capable of issuing an $\overline{\text{ERROR}}$ output at pin 26. The $\overline{\text{ERROR}}$ signal is inverted by pin 2 of U241 and is driven as interrupt request 13 to the interrupt controllers located on the I/O card. The interrupt controllers process the signal according to its priority and then signify the CPU of the interrupt request. The CPU acknowledges the interrupt and branches to a subroutine contained within memory.

The 80286 CPU and the 80287 numeric processor extension both receive their 12 MHz clock signals (CLK286) from the control bus. Various other signals including $\overline{\text{RDY286}}$, RST287, chip select 80287 ($\overline{\text{NPS1}}$), and $\overline{\text{RESET}}$ are all transmitted over the CPU_CTRL bus.

Table 13-3 and Figure 13-6 define the signal names and device pinout of the 80287 numeric processor extension.

Table 13-3: 80287 Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 – 3	$\overline{\text{S1}}$, $\overline{\text{S0}}$, COD/$\overline{\text{INTA}}$	These inputs allow the numeric processor extension to monitor the execution of ESCAPE instructions by the 80286 CPU.
5 – 8, 11, 12, 14 – 23	D15 – D0	This 16-bit bidirectional data bus interfaces directly to the 80286 16-bit data bus. See Figure 13-6.
9	VCC	+ 5 VDC power supply.
10, 30	VSS	System ground.
24	PEREQ	This numeric processor extension request indicates to the 80286 CPU that it is ready to execute a numeric operation.
25	$\overline{\text{BUSY}}$	This signal is asserted by the numeric processor extension to indicate to the CPU that it is currently executing a numeric operation.

Table 13-3 (continued): 80287 Device Signal Names

26	$\overline{\text{ERROR}}$	This signal indicates that an unmasked error condition exists.
27	$\overline{\text{NPRD}}$	This numeric processor extension read signal enables transfer of data from the 80287.
28	$\overline{\text{NPWR}}$	This numeric processor extension write signal allows transfer of data to the 80287.
29, 31	CMD0, CMD1	These command lines, along with select inputs, allow the 80286 to direct operations to the numeric processor extension.
32	CLK	This clock input provides the fundamental timing for internal 80287 operations. Special MOS level inputs are required.
33, 34	NPS2, $\overline{\text{NPS1}}$	These numeric processor extension select signals are asserted when the CPU is executing an ESCAPE instruction. If $\overline{\text{NPS1}}$ is low, and NPS2 is high, the 80287 can perform floating point instructions.
35	RESET	System reset. Causes the 80287 to terminate its current program execution and enter a dormant state.
36	$\overline{\text{PEACK}}$	This signal acknowledges to the 80287 that its request signal (PEREQ) has been recognized. If there are no more numeric operations pending, the request signal will be withdrawn.
37	CLK286	This input provides a sampling edge for the 80287 $\overline{\text{S1}}$, $\overline{\text{S0}}$, COD/$\overline{\text{INTA}}$, $\overline{\text{READY}}$, and HLDA inputs. This line must be connected to the 80286 CLK input.
38	HLDA	This input informs the 80287 when the CPU controls bus activity. It must be connected to the 80286 HLDA output.
39	CKM	This signal indicates whether the CLK input is to be divided by three or used directly.
40	$\overline{\text{READY}}$	The end of a bus cycle is indicated by this input. It must be connected to the 80286 $\overline{\text{READY}}$ input.

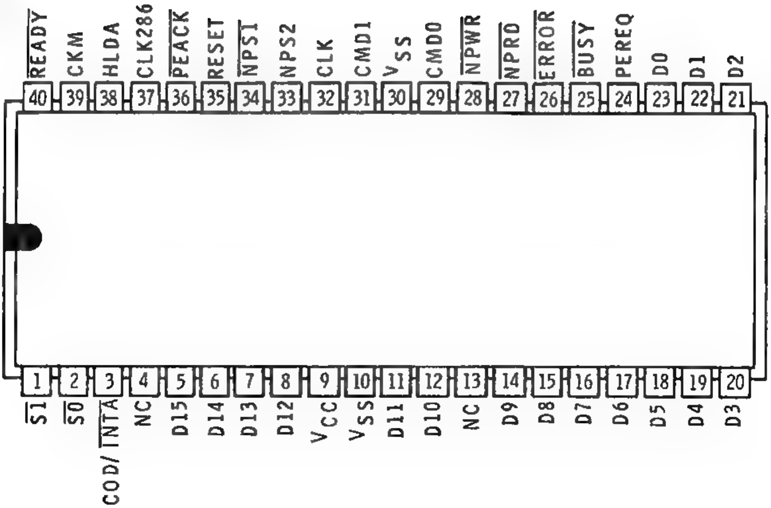


Figure 13-6: 80287 Device Pinout

Random Access Memory

Memory Organization

The system memory used by the 80286 microprocessor is arranged in an 8-bit byte format. Two bytes can be accessed to derive a word data segment. Either even or odd addresses can be assigned for byte or word data segments. Refer to Figure 13-7 for a system memory map.

NOTE: The shaded areas in Figure 13-7 do not indicate available memory. They differentiate the address boundaries that are used or reserved.

The standard configuration of the computer incorporates 512K of system memory (DRAM). The memory is arranged in even and odd addressed groups that can be selectively read from or written to through control logic. The $\overline{\text{MEMR}}$ (memory read) and $\overline{\text{MEMW}}$ (memory write) signals are used by the CPU during data transfer into or out of memory. Whether the CPU addresses the even or odd group of memory is determined by information on the address bus and enabled or disabled data transceivers residing on the data bus.

Memory locations are addressed over the DRAM_ADDR bus that carry the memory address MA0 – MA8 lines. These nine lines allow as much as 512K of memory to be addressed. When a valid memory location is addressed, data can be written to or read from that location. The $\overline{\text{WE}}$ (write enable) signal determines which operation will take place. Data is transferred into and out of DRAM over the MEM_DATA bus.

There are two banks of EPROM (erasable programmable read-only memory) located on the CPU/RAM card. Each bank contains two 32K × 8 devices. EPROM memory locations are addressed over a 16-bit address bus (MEM_ADDR). The chip select signals also are carried over this bus. These "monitor" ROMs contain information such as diagnostics and error messages. In addition, 4K of SRAM (special scratchpad RAM) exists on the CPU/RAM card for use by the CPU during ROM based functions.

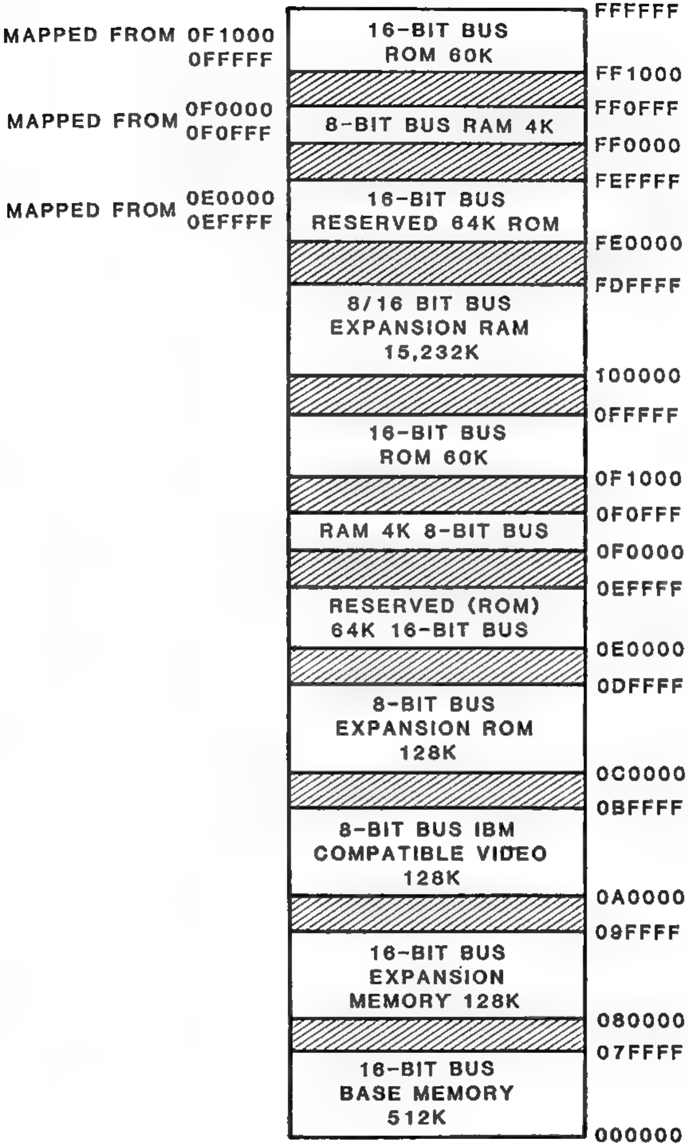

MAPPED FROM 0F1000 0FFFFF
MAPPED FROM 0F0000 0F0FFF
MAPPED FROM 0E0000 0EFFFF
16-BIT BUS ROM 60K
8-BIT BUS RAM 4K
16-BIT BUS RESERVED 64K ROM
8/16 BIT BUS EXPANSION RAM 15,232K
16-BIT BUS ROM 60K
RAM 4K 8-BIT BUS
RESERVED (ROM) 64K 16-BIT BUS
8-BIT BUS EXPANSION ROM 128K
8-BIT BUS IBM COMPATIBLE VIDEO 128K
16-BIT BUS EXPANSION MEMORY 128K
16-BIT BUS BASE MEMORY 512K
FFFFFF
FF1000
FF0FFF
FF0000
FEFFFF
FE0000
FDFFFF
100000
0FFFFF
0F1000
0F0FFF
0F0000
0EFFFF
0E0000
0DFFFF
0C0000
0BFFFF
0A0000
09FFFF
080000
07FFFF
000000

Figure 13-7: System Memory Map

Memory Address Bus

Refer to Figure 13-8 for the schematic of the memory address bus circuit.

The system ADDR_BUS is interfaced to edge connectors on the backplane, octal buffers U267 and U268, and to logic array address decoder U266. The addresses driven across the backplane allow the CPU to initiate communication with other cards present in the system. Octal buffer/drivers U267 and U268 are permanently enabled with their G inputs tied to ground. The output of these devices produce the memory address (MEM_ADDR) bus that addresses system EPROM and SRAM (special scratchpad RAM).

Logic array U266 is programmed as a decoder to derive the following control signals: $\overline{\text{MSEL}}$ (memory select) which used in conjunction with $\overline{\text{SA0}}$, determines whether the low bank, high bank, or both banks of RAM are accessed. The $\overline{\text{XRAM}}$ signal is used to allow the SRAM (special scratchpad RAM) to be read or written to. The $\overline{\text{RAM}}$ signal is used to derive the $\overline{\text{FAST}}$ (fast memory access) signal. The $\overline{\text{ROM}}$ signal enables the system ROM to be read by the CPU. The MX8 signal is driven to the DRAM_ADDR bus where it becomes the MA8 (memory address 8) line. The $\overline{\text{LMEG}}$ (low megabyte) signal is used in conjunction with the $\overline{\text{MEMR}}$ and $\overline{\text{MEMW}}$ signals to produce the $\overline{\text{SMEMR}}$ (memory read) and SMEMW (memory write) signals.

Memory Control Logic

Refer to Figure A-5 in Appendix A for the schematic of the memory control logic circuit.

The RAM portion of the CPU/RAM card is normally populated with 512 kilobytes plus two parity bits for each word (16-bits). Memory control logic induces correct timing generations, memory refresh, and parity generation and checking. It also generates specific read and write commands to either ROM or RAM banks. Memory control logic defines addresses for each respective source of memory (ROM or RAM) and interfaces the data bus (MEM_DATA) to the correct source of memory so the CPU can transfer data.

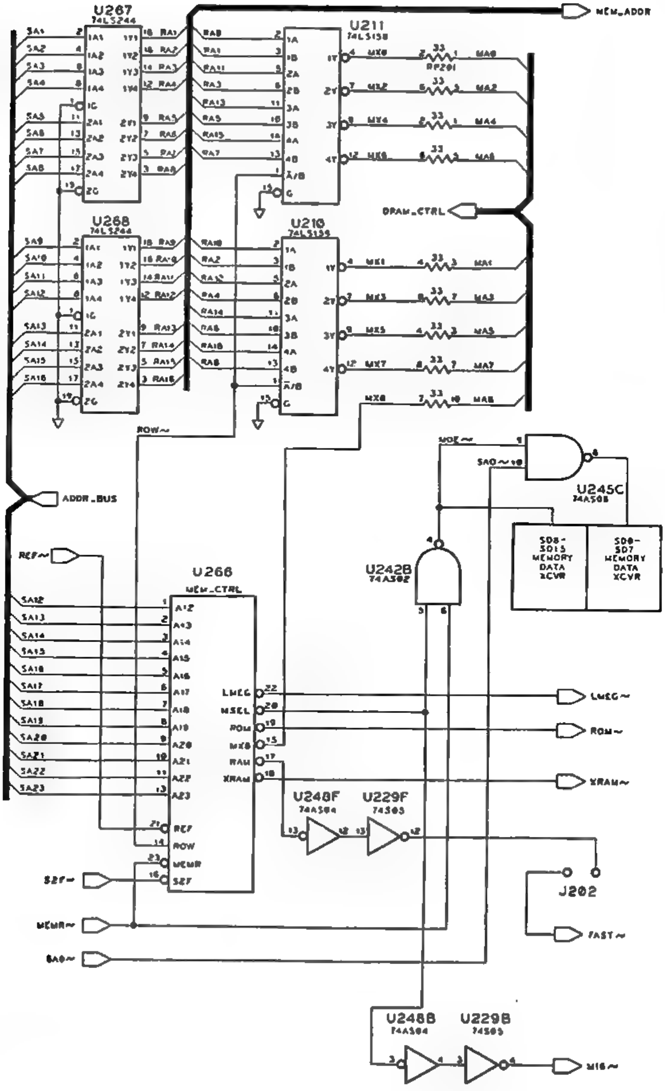


Figure 13-8: Memory Address Bus Schematic

A logic array decodes addresses from the SA12 – SA23 address bus (ADDR_BUS) to produce required control signals. Addresses on the SA1 – SA16 portion of the ADDR_BUS are buffered to form the memory address (MEM_ADDR) bus that addresses system EPROM and SRAM (static scratchpad RAM). Data also is multiplexed off the MEM_ADDR bus to create the memory address lines that address system DRAM (MA0 – MA8). The MA0 – MA8 address lines are routed to system DRAM as the DRAM_ADDR bus.

Logic array U266 decodes address and status signals used during specific operations. The $\overline{\text{MSEL}}$ (memory select) signal in conjunction with the SA0 signal, determine whether the low bank, high bank, or both banks of RAM are accessed. The SA0 signal in conjunction with the $\overline{\text{WRT}}$ signal is used to allow the CPU to read or write data to or from the low bank of DRAM. The system DRAM bus high enable (SBHE) signal is used in conjunction with the $\overline{\text{WR}}$ (write) signal to allow the high bank of DRAM to be accessed. The read and write signals and the system memory refresh signals are gated together to form the DRAM control bus (DRAM_CTRL).

Data can be transferred either to or from the low or high bank of system DRAM by data transceivers. To write data into DRAM, the $\overline{\text{WR}}$ (write) and $\overline{\text{MEMW}}$ (memory write) signals must both be true. The 8-bit data transceivers (U234 and U236) are then enabled to transfer data from the DATA_BUS to the MEM_DATA (memory data) bus, and therefore into DRAM, provided that a valid address has accessed a memory location.

To read data from a memory location, control logic is used to allow either the low (8-bit) or high (8-bit) bank of DRAM to be read, or both (16-bits). The $\overline{\text{SA0}}$ signal is ANDed with the MOE (memory output enable) signal at U245 to provide the low byte data transceiver with the LO_RD_EN (low read enable) signal. The active MOE signal enables the high bank of DRAM to be read.

As mentioned, the memory control logic contains parity generation and checking circuitry. U237 is a parity generator/checker that works on the low order memory bank, and U235 is a parity generator/checker that works on the high order bank.

Byte Data Access

Bits D0 – D7 of the CPUDATA bus are used to access even addressed memory locations and bits D8 – D15 access odd-addressed memory locations. The status of the $\overline{\text{SA0}}$ in conjunction with the status of the $\overline{\text{SBHE}}$ (bus high enable) signal from the CPU, determine which area of memory (even or odd) to access. When either bank is selected, bits A1 – A19 of the SA0 – SA23 address bus select a particular location within memory (real address mode).

Word Data Access

16-bit words are accessed by enabling both memory banks contiguously. The least significant 8-bit byte is located in even-addressed memory and the most significant byte in odd-addressed memory. The address bus selects the least significant byte from an even-addressed memory location, and then the most significant byte from an odd-addressed memory location.

If a word begins at an odd address, the CPU recognizes the fact and performs two accesses. In this case, the least significant byte is transferred across bits D8 – D15 of the data bus. The address is incremented by 1 and the most significant byte is transferred across bits D0 – D7 of the data bus. This operation automatically occurs whenever a word is accessed beginning with an odd address.

DRAM Refresh

Each memory address location within the DRAM memory banks must receive refresh pulses at least every 4 ms so that data integrity is not lost. The $\overline{\text{RAS}}$ (row address strobe) implements this operation.

During the refresh cycle, the CPU control logic drives the $\overline{\text{MEM}}$ signal true. This signal initiates the $\overline{\text{RAS}}$ signal by driving pin 6 of U245 true. The $\overline{\text{MEM}}$ signal is also applied to a TTL delay line that drives the $\overline{\text{ROW}}$ signal (pin 12 of U250) true 29 ns later. The $\overline{\text{ROW}}$ signal enables memory address multiplexers U210 and U211 to multiplex memory addresses onto the MA0 – MA7 address lines. Logic array U266 provides the MX8 signal that becomes the MA8 address line. The MA0 – MA8 address lines form the DRAM_ADDR (DRAM address) bus. Binary counter U271 generates refresh addresses to the DRAM banks so that the $\overline{\text{RAS}}$ strobe is able to access memory and revitalize it. Binary counter U271 is enabled when the true $\overline{\text{REF}}$ signal is asserted.

The $\overline{\text{REF}}$ signal also indicates to logic array U266 that it should drive its outputs inactive during a refresh cycle. This arbitration logic prevents the CPU from accessing RAM during refresh.

The $\overline{\text{RAS}}$ signal causes the low order memory address to be latched into DRAM and the $\overline{\text{CAS}}$ (column address strobe) signal causes the high order address to be latched. These signals are transmitted over the DRAM_CTRL bus along with the $\overline{\text{WE0}}$ and $\overline{\text{WE1}}$ signals.

Data Bus Interface

The low order data byte (D0 – D7) is transferred between the DATA_BUS and the MEM_DATA bus through transceiver U236. The active LO_RD_EN signal allows data (D0 – D7) to be read from DRAM. The $\overline{\text{WRT\_EN}}$ signal allows data to be written into DRAM. The MOE (memory output enable) signal allows the CPU to read data bits D8 – D15 from DRAM and the $\overline{\text{WRT\_EN}}$ signal allows a write operation to be executed. By enabling either, or both data transceivers, the CPU can transfer either byte or word information respectively, into and out of DRAM.

Chapter 14
Input/Output Card

The following sections describe the function of the I/O (input/output) card. A simplified block diagram of the I/O card is illustrated in Figure A-6 of Appendix A. Refer to this block diagram during the following discussions in this chapter.

Specifications

Serial Port: Serial communication through a 9-pin "D" type connector.

Parallel Port: Parallel communication through a 25-pin "D" type connector.

Keyboard Interface: Bidirectional communication through a 5-pin DIN connector on the backplane; processed by a system control processor on the I/O card.

Real-Time Clock: Provides a time-of-day clock with an alarm, and a calendar that covers a span of 100 years.

Interrupt Controllers: Two cascaded 8259 programmable interrupt controllers prioritize interrupt requests from as many as 15 different devices.

DMA Controllers: Two cascaded 8237A-5 DMA controllers to process 8-bit or 16-bit DMA data transfers.

Address Decode Logic

Refer to Figure A-7 in Appendix A for the schematic of the address decode logic circuit.

The address decode circuit receives I/O port addresses on the SA0 – SA9 address lines. The port addresses and the I/O commands $\overline{\mathrm{XIOR}}$ and $\overline{\mathrm{XIOW}}$ are decoded into the appropriate chip select or port select signals by logic array U341.

Logic array U341 transforms signals at its inputs into the parallel port control signals. The $\overline{\mathrm{RPPA}}$ (read parallel port A), $\overline{\mathrm{RPPB}}$ (read parallel port B), $\overline{\mathrm{RPPC}}$ (read parallel port C), $\overline{\mathrm{WPPA}}$ (write parallel port A), and $\overline{\mathrm{WPPC}}$ (write parallel port C) signals control parallel communications with a peripheral device through the parallel port. U341 also decodes the $\overline{\mathrm{CS50}}$ (serial port chip select) signal to enable asynchronous communications controller U315 to control serial communication with a peripheral device through the serial port.

The $\overline{\mathrm{LED}}$ signal clocks the diagnostic LED flip-flop (U303) as the power-up self tests are executed. When a test is successfully completed, the $\overline{\mathrm{LED}}$ signal is asserted low to cause the output of the flip-flop that drives the LED (corresponding to the particular test) to change states. As this occurs, the LEDs go out sequentially for each successful test.

The $\overline{\mathrm{XD245d}}$ (data direction) signal controls the I/O card data bus buffer (U354). When this signal is true, data can be transferred from the I/O card to the system data bus.

Whenever a DMA cycle is in progress, the $\overline{\mathrm{IOAEN}}$ (I/O address enable) signal is driven true to disable all chip select signals to devices that normally use the system busses. During a DMA transfer, the DMA controller must have control of the system busses. If any other device were to use the bus during a DMA cycle, data on the bus could be destroyed.

If the Z-419 bit-mapped video card is installed in the computer and the Z-100 Mode DOS has been booted, the 319EN signal is asserted to disable the chip select signal to the 16-bit DMA controller. This is done to avoid address clashes with other cards in the system and because the Z-419 does not require use of the upper eight bits (D8 – D15) of the bus.

The $\overline{\text{XIOR}}$ and $\overline{\text{XIOW}}$ signals applied to logic array U341, instruct the device to output data to support either a read or write operation to or from the parallel port. If the CPU receives an interrupt request from either PIC (programmable interrupt controller), the $\overline{\text{INTA}}$ (interrupt acknowledge) signal is applied to U341. When the logic array recognizes the $\overline{\text{INTA}}$ signal, it drives its $\overline{\text{XD245D}}$ output low. This allows the CPU to read the vector address provided by the PIC. The CPU then branches to the specified memory location and executes instructions found there.

U340 is a 3-to-8 line decoder that uses data on the ADDR_BUS in conjunction with the output of U324 to derive chip select signals for other various devices used in the computer.

Serial Port

Refer to Figure A-8 in Appendix A for the schematic of the serial port circuit.

Address decode logic provides the serial port controller with its chip select signal ($\overline{\text{CS50}}$) I/O read/write control signals ($\overline{\text{XIOR}}$ and $\overline{\text{XIOW}}$). The serial port is located at I/O address 3F8 – 3FF (COM1) or I/O address 2F8 – 2FF (COM2), and allows the CPU to communicate with peripheral devices that recognize serial data signals. Data transmission between computer and data terminal equipment (DTE) is interfaced through a 9-pin connector. Refer to Table 14-1 and Figure 14-1.

Table 14-1: **Serial Connector Pinout and Signal Description**

DB-9 PIN NUMBER	SIGNAL	DESCRIPTION	DATA FLOW	DB-25 PIN NUMBER
1	CD	Carrier detect	In	8
2	SIN	Receive data	In	3
3	SOUT	Transmit data	Out	2
4	DTR	Data terminal ready	Out	20
5	GND	Signal ground	—	7
6	DSR	Data set ready	In	6
7	RTS	Request to send	Out	4
8	CTS	Clear to send	In	5
9	RI	Ring indicate	In	22
Case	GND	Chassis ground	—	1

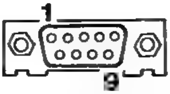


Figure 14-1: Serial Port Connector

Received signals are buffered and converted to TTL levels by U319 and U334. The signals are applied to an NS16450 programmable asynchronous communications controller that controls the activities of the serial port. The device pinout of the NS16450 is illustrated in Figure 14-2.

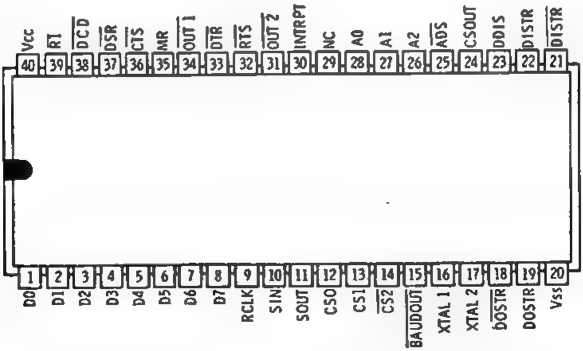


Figure 14-2: Serial Port Controller Device Pinout

When serial data is transmitted from a peripheral to the computer, the communications controller converts the serial data stream into an 8-bit byte. The assembled byte is stored in a temporary register until the CPU executes a read operation over the XD0 – XD7 data bus. Data on the XD0 – XD7 data bus is interfaced to the CPU_DATA bus (when the CPU reads the assembled byte) through transceivers and buffers.

When the computer transmits data to a peripheral device, the CPU places data onto the XD0 – XD7 data bus (through buffers and transceivers) in byte format. The communications controller converts the data into serial format in preparation for transmission to a peripheral device. Transmitted signals are buffered and inverted by U318, then filtered by LC networks to dampen RFI emissions.

The $\overline{\text{CS50}}$ signal is used to enable the communications controller while the XSA0, XSA1, and SA2 signals are used to address its internal registers. The status of the $\overline{\text{XIOR}}$ and $\overline{\text{XIOW}}$ signals inform the communications element whether the CPU wishes to read from or write into the port.

Y301 is a 1.8432 MHz oscillator that clocks the communication controller's internal registers and baud rate generator.

The serial port interrupt request may be configured as IRQ3 (COM2) or IRQ4 (COM1) by placement of jumper J304. The configured interrupt signal is buffered and driven to the interrupt controllers through pin 8 of U316. The interrupt controllers establish the priority of the interrupt request and drive the INTR (maskable interrupt request) signal to the CPU to cause the CPU to recognize and service the request being made by the serial device.

Parallel Port

Refer to Figure A-9 in Appendix A for the schematic of the parallel port circuit.

The I/O card interfaces parallel data from the computer to an external parallel device, and status signals from the device to the computer. The entire parallel port is actually comprised of three individual ports. Port B is a read-only port that drives status signal data from the parallel device onto the parallel data bus. Ports A and C are read and write ports. Port A is the parallel data port and port C the device command port. The read ports associated with A and C are used to perform a check of the data sent out to the parallel device, to make sure, for example, that false data is not present on the data lines as a result of malfunctions in the device.

The interrupt request for the parallel port may be configured as either IRQ5 or IRQ7. The status of the $\overline{\text{ACK}}$ signal is used to trigger the interrupt request signal. The status of the signal at pin 12 of U336 is inverted to enable 3-state buffer U304 to conduct the interrupt request signal to jumper J303. Shorting pins 1 and 2 of J303 configure the interrupt request as IRQ7 (LPT1), and shorting pins 2 and 3 configure the interrupt request as IRQ5 (LPT2). The interrupt request signal causes the CPU to recognize and service the request being made by the parallel device.

A 25-pin connector interfaces data and control signals between computer and peripheral (parallel) device. This connector extends through the rear of the computer. Table 14-2 defines the signals interfaced through the parallel 25-pin connector (P303). The FLOW column indicates the direction of signal travel with respect to the attached parallel device. Figure 14-3 illustrates the pinout of the connector.

Table 14-2: **Parallel Connector Pinout**

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	$\overline{\text{STROBE}}$	Gates data from external device.
2 – 9	DATA0 – DATA7	Parallel data bits 0 – 7. Active high.
10	$\overline{\text{ACK}}$	Printer acknowledgment of receipt of data and ready to accept more data. Active low.
11	BUSY	Prevents transmission of further data to printer when high, during data transfer, printing, off line, or in error state.
12	PE	When high, printer is out of paper.
13	SLCT	This active high signal indicates printer acknowledgment of selected operating state.
14	$\overline{\text{AUTO FD XT}}$	When low, automatically line feeds printer at end of a line.
15	$\overline{\text{ERROR}}$	Goes low when printer is in an error status, as when out of paper or off line.
16	$\overline{\text{INIT}}$	Resets printer controller to its initial conditions and clears its data buffers.
17	$\overline{\text{SLCT IN}}$	When low, allows data to be placed in the printer's data buffer.
18-25	GND	Ground.

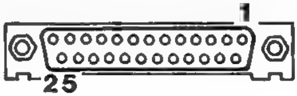


Figure 14-3: Parallel Port Connector

Port Decoder

Logic array U341 (Figure A-7 in Appendix A) decodes the signals required for use by the parallel port. The status of signals XSA0 and XSA1 indicate to the logic array which port is to be active. Refer to Table 14-3.

Table 14-3: Parallel Port Selection

XSA1	XSA0	PORT
0	0	A
0	1	B
1	0	C

Data Transfer

Logic array U341 decodes the port address and the I/O commands ($\overline{\text{XIOR}}$ and $\overline{\text{XIOW}}$) to generate the appropriate port select (A, B, or C) signal. This logic array automatically decodes these signals. An active output from the logic array enables the respective port (A, B, or C) and initiates a read or write operation through the parallel port. If any of the outputs of the logic array are inactive, the devices connected to the signal are also inactive.

Writing to Port A

Port A is located at address 378 or 278. The $\overline{\text{WPPA}}$ signal controls the write port A function. At the beginning of the write cycle, $\overline{\text{WPPA}}$ is driven low. This allows data on the XD0 – XD7 interface bus to stabilize. On the next low-to-high transition of the $\overline{\text{WPPA}}$ signal, octal D-type latch U333 gates data on the XD0 – XD7 interface bus to its "Q" outputs. The latched data is written to the peripheral parallel device as soon as it is latched. The data appears at the output of U333, and is routed to the diagnostic read bus driver for port A.

Inductors L305 – L312 and capacitors C342, C343, C348, C355, C356, C370, C371, and C372 stabilize the output signals and minimize noise during transmission. The signals are transmitted through a 25 pin D-type connector (P303) to the peripheral device.

Reading Port A

When the active low $\overline{\text{RPPA}}$ signal is asserted, it causes buffer/driver U332 to transfer the status of the DATA0 – DATA7 (peripheral parallel data) signals interfaced through P303, onto the XD0 – XD7 data bus. The DATA0 – DATA7 signals are read by the CPU through various buffers and transceivers. CPU diagnostics compare this data with the data sent out and, if different, indicate an error condition.

Reading Port B

Port B is located at address 379 or 279. When the active low $\overline{\text{RPPB}}$ signal is asserted, it enables one-half of buffer/driver U331 to transfer the status of the SLCT, PE, $\overline{\text{ACK}}$, and BUSY signals onto data bus lines XD4 – XD7. The $\overline{\text{RPPB}}$ signal also enables 3-state buffer U326 to conduct the status of the $\overline{\text{ERROR}}$ signal onto data bus line XD3.

Writing to Port C

Port C is located at address 37A or 27A. At the beginning of the write cycle, the $\overline{\text{WPPC}}$ signal is low, allowing data on the XD0 – XD7 bus to stabilize. On the next low-to-high transition, flip-flop U336 latches the status of the signals present on the XD0 – XD7 data bus to its "Q" outputs. The signals are buffered and driven by U335 as the $\overline{\text{STROBE}}$, $\overline{\text{AUTO\_FD\_XT}}$, $\overline{\text{INIT}}$, and $\overline{\text{SLCT\_IN}}$ signals to the peripheral (parallel) device. Inductors L301 – L304 and capacitors C319, C320, C326, and C340 stabilize output signals and minimize noise during transmission. Pull-up resistors contained within resistor pack RP305 are used on the write port C transmission lines.

Reading Port C

When the active-low RPPC signal is asserted, it enables one-half of buffer/driver U331 to transfer the status of the STROBE, AUTO_FD_XT, INIT, and SLCT_IN signals onto data bus lines XD0 – XD3. The RPPC signal also enables 3-state buffer U326 (pin 3) to conduct the status of the signal at pin 12 of U336 onto data bus line XD4. CPU diagnostics check the logic levels of these signals, compare them to the data sent out and, if different, indicate an error condition.

Reset

The IORST (I/O reset) signal is applied to the clear input of the write port C data latch U336. When this signal is active, the status of the signals at the outputs of U336 are reset.

Programmable Interrupt Controllers

Refer to Figure A-10 in Appendix A for the schematic of the interrupt control circuit.

There are two cascaded 8259 PIC (programmable interrupt controller) devices on the I/O card. Any of 15 possible devices can generate a maskable interrupt request to the PICs. The controllers are programmed to prioritize the interrupt requests, then output one request signal (INTR) to the CPU. The CPU can tell the 8259 to ignore (mask) all interrupts or change the priority of the interrupts. The device pinout of the 8259 interrupt controller is illustrated in Figure 14-4.

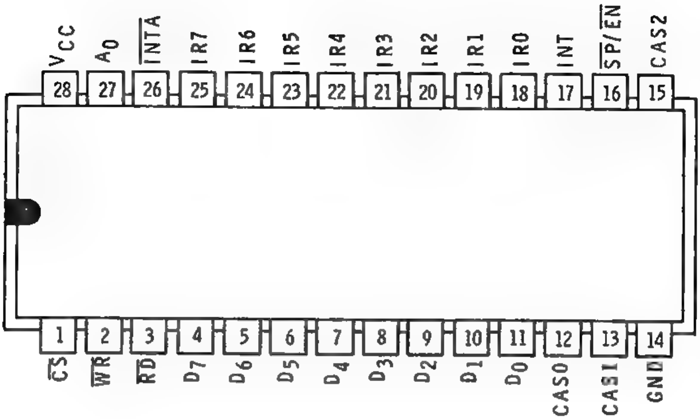


Figure 14-4: Programmable Interrupt Controller Device Pinout

When the system is powered up, the CPU must initialize the PIC by providing it with a control word. The initialization process is as follows: The respective PIC is enabled (low on pin 1), the control word is placed on the data bus, then the $\overline{\text{XIOW}}$ (write) pulse is brought low (pin 2). The written control word clears internal registers and sets the priority of the interrupts. A control word can be written at any time to clear the registers or to change the order of priorities.

The $\overline{\text{CS59\_1}}$ signal enables interrupt controller U353 and the $\overline{\text{CS59\_2}}$ signal enables interrupt controller U350. The 15 independent interrupt requests are divided into two 8-bit bytes that are each applied to one PIC. IRQ0 – IRQ7 are processed by U353 and IRQ8 – IRQ15 are processed by U350. When one of the interrupt inputs goes high, the device sets and checks its interrupt request register. If an interrupt is in progress, the 8259 will wait until the current interrupt is completed and then begin the process.

When the high order byte interrupt controller (U350) receives an interrupt request from a device, it processes the request in terms of its priority and then outputs an interrupt request signal to the IRQ2 input of U353. The interrupt request then goes through another prioritizing sequence before PIC U353 drives an active interrupt request (INTR) to the CPU.

The CPU responds by placing a low on the $\overline{\text{INTA}}$ (interrupt acknowledge) line after it completes its current instruction. The 8259 checks its three cascading inputs and when a second $\overline{\text{INTA}}$ signal comes from the CPU, the "Type Number" or interrupt instruction vector address is sent to the CPU on the data lines. The CPU uses the vector address to branch to a subroutine and execute instructions found there. When the last instruction in the interrupting routine is completed, an end of interrupt command is sent to the 8259. The PIC interrupt request register is cleared after completion of the second $\overline{\text{INTA}}$ cycle.

The XSAO signal allows the registers within the PICs to be addressed, and the true $\overline{\text{XIOR}}$ and $\overline{\text{XIOW}}$ signals allow the CPU to transfer data into or out of registers within either PIC.

Programmable Interval Timer

Refer to Figure A-11 in Appendix A for the schematic of the programmable interval timer circuit.

The PIT (programmable interval timer) generates precise time delays into the CPUs program execution. The 8254 contains three 16-bit programmable timers that allow for great flexibility in matching the device to system requirements. All modes of operation are software programmable through control signals and data provided by the CPU. An internal 3-state buffer connects the PIT to the XD0 – XD7 data bus. The counter outputs are OUT0, OUT1, and OUT2. The device pinout of the 8254 PIT is illustrated in Figure 14-5.

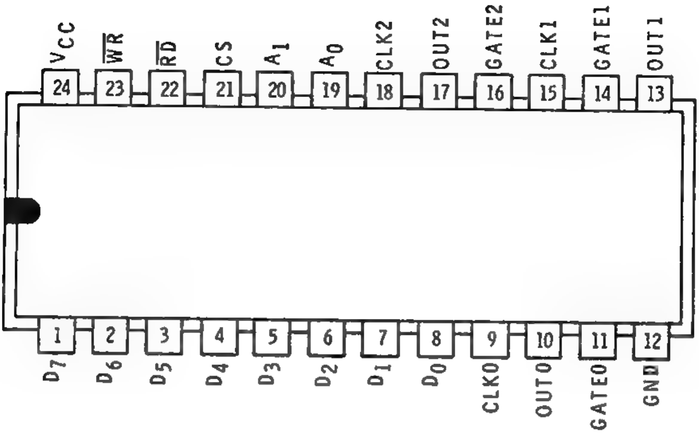


Figure 14-5: Programmable Interval Timer Device Pinout

The true $\overline{\text{CS54}}$ signal enables the programmable interval timer. Specific combinations of the $\overline{\text{XIOR}}$, $\overline{\text{XIOW}}$, XSA0, and XSA1 signals allow the CPU access to particular internal registers of the device so that the CPU can program the device. Data is transferred over the XD0 – XD7 data bus.

A 14.31818 MHz crystal oscillator (Y303) provides the fundamental timing signal to the interval timer. This signal also is distributed to other parts of the computer as the OSC signal. Binary counter U355 divides the base frequency output of Y303 into a frequency of 1.193 MHz that clocks the CLK0, CLK1, and CLK2 lines of U345.

The 8254 can operate in a number of modes determined by a control word issued by the CPU. Refer to the following for a description of the modes in which the interval timer operates.

- Mode 0 – This mode produces a particular operation or event upon termination of a count sequence. A terminal count is loaded into a selected clock that begins to count (increment) from 0 (zero) until the predetermined value is reached. When the terminal count is reached, the OUT line of the interval timer is driven high. The active high signal may be used to trigger an event such as an interrupt, or it may drive an external device.

- Mode 1 – During this mode, the CPU provides the interval timer with a control word that enables an internal counter. The OUT line is initially high until a CLK pulse triggers it low. The OUT line will remain low until the counter terminates (reaches zero) and will then return to a high state.

- Mode 2 – This mode of operation provides a divide by "N" rate generator output. A control word is written into an internal register by the CPU. When the terminal count reaches 1, the OUT line will go low for the duration of one incoming clock pulse.

- Mode 3 – This mode of operation is very similar to mode 2. However, during mode 3, the duty cycle is different than that of mode 2. The OUT signal will remain high for one-half of the terminal count and then will go low. This produces a square wave output.

- Mode 4 – The OUT signal is normally high during this mode but switches low when the terminal count is reached. The duration of the low is for one clock pulse, then the OUT line goes high again. This mode is similar to mode 2.

- Mode 5 – This mode provides a hardware triggered strobe. The terminal count begins on a low-to-high transition on a GATE input and drives the OUT line low for one clock pulse when the terminal count is reached.

Table 14-4 indicates the programming instructions provided by the CPU that configure the 8254 for different mode operation.

Table 14-4: Programmable Interval Timer Programming Codes

CS	$\overline{RD}$	$\overline{WR}$	A1	A0	FUNCTION
0	1	0	0	0	Load counter 0.
0	1	0	0	1	Load counter 1.
0	1	0	1	0	Load counter 2.
0	1	0	1	1	Write mode word.
0	0	1	0	0	Read counter 0.
0	0	1	0	1	Read counter 1.
0	0	1	1	0	Read counter 2.
0	0	1	1	1	3-state.
1	X	X	X	X	3-state.
0	1	1	X	X	3-state.

X = Don't Care

The OUT0, OUT1, and OUT2 counters are loaded with a starting count and decrement until zero is reached. When a counter reaches zero, a signal is sent out of the related output. The OUT0 signal can be processed as a system interrupt request to cause a specific function to occur. The OUT0 signal is driven as the IRQ0 interrupt request to the interrupt controllers, the OUT1 signal is latched by flip-flop U360, and is used to generate a refresh cycle. Note that channel 2 should never be reprogrammed because refresh may be lost. The OUT2 signal is used to provide an audible frequency to the speaker. The audible signal is gated through U343 and driven by U301 through connector P301 to the speaker.

DMA Control

Refer to Figure A-12, A-13, A-14, and A-15 in Appendix A for the schematic of the DMA control circuit (parts 1, 2, 3, and 4).

Two cascaded 8237A-5 DMA controllers speed data transfer between memory and I/O devices. The primary advantage to DMA data transfer is that data can be moved without having to be passed through the CPU. The DMA controllers have built-in instruction sets that allow them to operate with greater speed.

When the CPU desires to transfer data, it specifies the address at which the data will be found, or the address at which the data is to be placed, to the DMA controller. The DMA controller takes control of the system busses and then executes its internal instruction set to transfer the data.

Each DMA controller is capable of generating up to 16 address bits over their A0 – A7 outputs. Alternate 8-bit address bytes are placed on the A0 – A7 bus to be latched to the SA0 – SA23 address bus when the $\overline{\text{DMA\_1}}$ signal is true. When an address has been specified, the DMA controller uses the system data bus to transfer data.

The DMA controllers generate all of the necessary memory read and memory write signals during a data transfer and can operate in one of three different modes: A single byte of data can be transferred, a block of data can be transferred with both the start and end of the block defined by the CPU, or a block of data can be transferred with the end of the block defined by an external signal on the EOP (end of process) line.

When the DMA is to control the bus, a hold request is issued to the CPU. The CPU responds with a hold acknowledge (HLDA) signal and then goes into a hold state until the transfer is completed. A peripheral can request DMA service over one of the DREQ (DMA request) lines. The chosen DMA controller responds to the request by initiating a CPU hold, and then sending a DMA acknowledge ($\overline{\text{XDACK}}$) signal to the peripheral. The device pinout of the 8237A-5 DMA controller is illustrated in Figure 14-6.

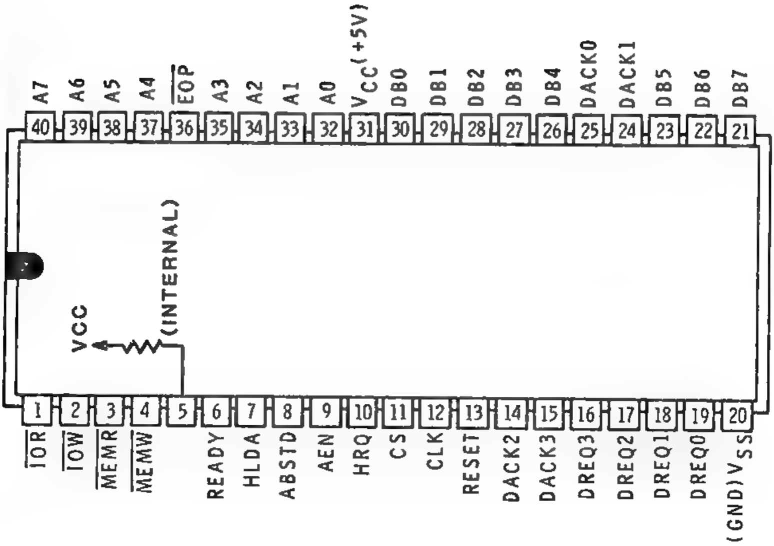


Figure 14-6: DMA Controller Device Pinout

8-Bit DMA Control

Logic array U310 decodes required control signals for timing and control of 8-bit DMA data transfers. When the CPU wants to use 8-bit DMA control it writes data into selected registers of DMA processor U313. The DMA processor is then programmed to execute the operation (data transfer) while the CPU is in a hold state. The CPU is capable of programming the 8-bit DMA processor to operate within defined modes such as block movement of data or 8-bit byte movement. It should be noted that block movement of data can cause the refresh signal to be lost. Therefore, block movement is not recommended.

8-bit peripheral devices requiring 8-bit DMA service must request the service over the DREQ0 – DREQ3 signal lines. When DMA processor U313 receives a request, it issues a hold request (HRQ37_1) to 16-bit DMA processor U311 over the DREQ4 line (cascade line). DMA processor U311 then issues a hold request (HRQ37_2) to logic array decoder U310. The signals present at the inputs of logic array U310 allow the decoder to ascertain when to assert a hold acknowledge signal.

When the HRQ37_2 (hold request) signal is active, U310 drives the hold acknowledge (HLDA37_2) signal to DMA processor U311. The DMA processor (U311) does not perform a data transfer at this time (because channel 4 is programmed to operate in the cascade mode); rather it drives the $\overline{\text{DACK4}}$ signal that is inverted by pin 10 of U305, to the hold acknowledge input (HLDA) of DMA processor U313. When the DMA processor (U313) receives the HLDA signal, it is then ready for a data transfer, and waits in this state until the READY line is activated by the DMA state machine logic array (U309). However, at this time, the DMA processor is not connected to any system busses.

When the CPU is free to allow wait states, state machine logic array U309 outputs signals that configure required logic devices (buffers and latches) to accommodate the coming 8-bit DMA data transfer. The DMA signal from pin 17 of U309 is gated with the $\overline{\text{MASTER}}$ signal by pin 3 of U338 to produce the $\overline{\text{PRAEN}}$ (page register address enable) signal. This signal enables the command buffer U363 and address buffer U352. During 8-bit transfers, the state generator also drives the active ready (RDY37_1) signal to pin 6 of 8-bit DMA processor U313.

The status of various other control signals produced by the state generator synchronize the $\overline{\text{MEMR37}}$ (memory read), $\overline{\text{MEMW37}}$ (memory write), $\overline{\text{IOR37}}$ (I/O read), and $\overline{\text{IOW37}}$ (I/O write) signals through NAND gates U343 and U361. These decoded signals are driven through buffer/driver U363 when $\overline{\text{PRAEN}}$ is active. The status of the signals indicate whether a data transfer should take place to or from an I/O device or memory.

Page register U346 is used to generate the most significant address bits (during DMA transfers) that indicate the correct address at which data should be transferred to or from. When the $\overline{\text{PRAEN}}$ signal is active, it causes buffer/driver U352 to drive the address generated by the page register onto the address bus (bits 17 – 23 for 16-bit transfers and bits 16 – 23 for 8-bit transfers). The page register (U346) can also be used to derive data bytes, that are driven onto the XD0 – XD7 data bus when the enable page register ($\overline{\text{ENPR}}$) signal and the S1F signal are active.

During the 8-bit DMA process, the $\overline{\text{A}}$ signal from pin 16 of U310 is driven active to prevent a 16-bit DMA cycle from occurring. This is done to make sure that bus contention does not occur during a data transfer.

The A0 – A7 address lines of the 8-bit DMA processor interface to the SA0 – SA7 ADDR_BUS through transceiver U314. The $\overline{\text{DMA\_1}}$ signal in conjunction with the $\overline{\text{AEN37\_1}}$ signal determine direction of data flow through the transceiver. This 8-bit DMA address transceiver (U314) drives the address bus during DMA transfers, and drives addresses to the DMA controller to allow access to its internal registers.

The 8-bit DMA D0 – D7 data lines can also be interfaced to the XD0 – XD7 data bus through transceiver U351. This transceiver buffers the data to DMA controller U313 when in program mode. The direction of the transfer is controlled by the S1F signal. When this signal is high, a read operation occurs and when it is low, a write operation occurs. The output enable is controlled by the $\overline{\text{DEN}}$ (data enable) signal.

16-Bit DMA Control

Logic array U310 decodes various command and control signals required for proper timing and control of 16-bit DMA data transfers. When the CPU wants to use 16-bit DMA control, it writes data into selected registers of DMA processor U311. The DMA processor is then programmed to execute the operation (data transfer) while the CPU is in a hold state. The CPU is capable of programming the 16-bit DMA processor to operate within defined modes such as block movement of data or 16-bit word movement. Note that block movement of data can cause the refresh signal to be lost. Therefore, block movement is not recommended.

16-bit peripheral devices requiring 16-bit DMA service must request the service over the DREQ5 – DREQ7 signal lines. When DMA processor U313 receives a request, it issues a hold request (HRQ37_2) to logic array decoder U310. The signals present at the inputs of logic array U310 allow the decoder to ascertain when to assert a hold acknowledge signal. When the HRQ37_2 signal is active U310 drives the hold acknowledge (HLDA37_2) signal to DMA processor U311. Upon receipt of the hold acknowledge signal DMA processor U311 outputs the DMA acknowledge ($\overline{\text{DACK5}}$ – $\overline{\text{DACK7}}$) signal and waits for the DMA state machine logic array to activate the READY line.

When the CPU is free to allow wait states, state machine logic array U309 outputs signals that configure required logic devices (buffers and latches) to accommodate the coming 16-bit DMA data transfer. The DMA signal from pin 17 of U309 is gated with the $\overline{\text{MASTER}}$ signal by pin 3 of U338 to produce the $\overline{\text{PRAEN}}$ (page register address enable) signal. This signal enables the command buffer U363 and address buffer U352. During 16-bit transfers, the state generator also drives the active ready (RDY37_2) signal to pin 6 of DMA processor U311.

The status of various other control signals produced by the state generator synchronize the $\overline{\text{MEMR37}}$ (memory read), $\overline{\text{MEMW}}$ (memory write), $\overline{\text{IOR37}}$ (I/O read), and $\overline{\text{IOW37}}$ (I/O write) signals through NAND gates U343 and U361. These decoded signals are driven through buffer/driver U363 when $\overline{\text{PRAEN}}$ is active. The status of the signals indicate whether a data transfer should take place to or from an I/O device or memory.

Page register U346 is used to generate the most significant address bits (during DMA transfers) that indicate the correct address at which data should be transferred to or from. When the $\overline{\text{PRAEN}}$ signal is active, it causes buffer/driver U352 to drive the address generated by the page register onto the address bus (bits 17 – 23 for 16-bit transfers and bits 16 – 23 for 8-bit transfers). The page register (U346) can also be used to derive data bytes, that are driven onto the XD0 – XD7 data bus when the enable page register ($\overline{\text{ENPR}}$) signal and the S1F signal are active.

During the 16-bit DMA process, the $\overline{\text{B}}$ signal from pin 17 of U310 is driven active to prevent an 8-bit DMA cycle from occurring. This is done to make sure that bus contention does not occur during a data transfer.

The A0 – A7 address lines of the 16-bit DMA processor interface to the SA0 – SA7 ADDR_BUS through transceiver U312. The $\overline{\text{DMA\_2}}$ signal in conjunction with the $\overline{\text{AEN37\_2}}$ signal determine direction of data flow through the transceiver. This 8-bit DMA address transceiver (U314) drives the address bus during DMA transfers, and drives addresses to the DMA controller to allow access to its internal registers.

The 16-bit DMA D0 – D7 data lines can also be interfaced to the XD0 – XD7 data bus through transceiver U349. This transceiver buffers the data to DMA controller U311 when in program mode. The direction of the transfer is controlled by the S1F signal. When this signal is high, a read operation occurs and when it is low, a write operation occurs. The output enable is controlled by the $\overline{\text{DEN}}$ (data enable) signal.

Keyboard Interface

Refer to Figure A-16 in Appendix A for the schematic of the keyboard interface circuit.

An 8-bit 8042 UPI (universal peripheral interface) allows the 80286 CPU to communicate with the keyboard. The 8042 is a microprocessor that has internal ROM (2K × 8), RAM (1K × 8), I/O ports, and a clock generator built into it. The 8042 is programmed to handle bidirectional synchronized serial communications with the keyboard. The device pinout of the 8042 microcomputer is illustrated in Figure 14-7.

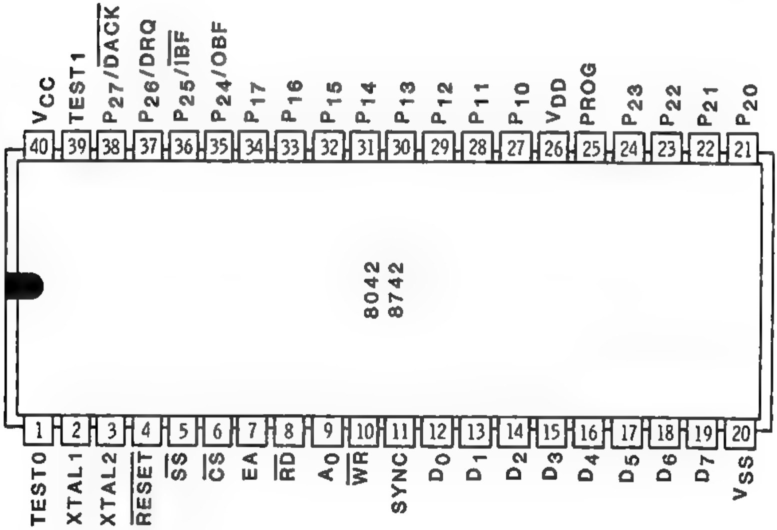


Figure 14-7: Keyboard Interface Device Pinout

Port 1 (P10 – P17) is programmed as an input port to interpret the configuration of various portions of the computer. This port detects the size of on-board RAM, the default display adapter, and the status of the keyboard inhibit switch.

Port 2 (P20 – P27) is an output port that can reset the system under software control. It also outputs data (KBDATA) and clock signals (KBCLK) to the keyboard.

The keyboard transmits data in a serial format. When data is received, an interrupt request (IRQ1) is driven to the PIC (programmable interrupt controller) devices. The PIC devices prioritize the interrupt request and then signify the CPU by driving the INTR (interrupt request) line true. KBDATA (keyboard data) transmission is synchronous with the KBCLK (keyboard clock) signal. KBDATA bits are clocked into testable input T0 on the active edge of the KBCLK signal. As KBDATA bits are received, the 8042 converts the transmission into bytes. The assembled byte is stored in an output buffer for the CPU to read.

Real-Time Clock Interface

Refer to Figure A-17 in Appendix A for the schematic of the real-time clock interface circuit.

The MC146818A real-time clock interface provides a time-of-day clock with an alarm, and a calendar that covers a span of one hundred years. The device is a low power CMOS IC that is ideal for use with a battery backup circuit. The device pinout of the MC146818A is illustrated in Figure 14-8.

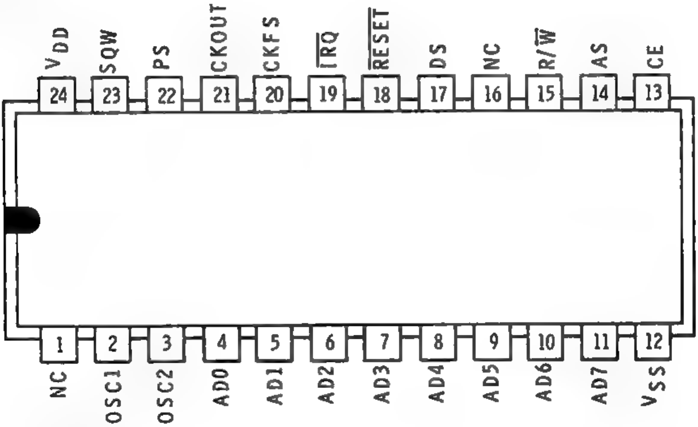


Figure 14-8: Real-Time Clock Chip Device Pinout

The MC146818A is located at I/O address port 70 and is enabled when the $\overline{\text{RTC}}$ signal is low. The $\overline{\text{XIOR}}$ and $\overline{\text{XIOW}}$ signals are used when the device is active to allow the CPU to transfer data into or out of the device over the XD0 – XD7 data bus.

This real-time clock can be programmed to provide periodic interrupt requests (IRQ8) or it can be used to trigger an event. The device receives its time base from a 32.768 kHz oscillator. The timing signals allow internal registers that log the time of day and calendar, to function. The B5V line provides the real-time clock with power (+5V) during normal operation and is switched to the battery backup (+3.5V) when power is switched off.

Chapter 15
Keyboard

Specifications

Keys: 84 keys with extended function capabilities.

LEDs: LED indicators on NUM LOCK, CAPS LOCK, and SCROLL LOCK keys.

Controller: 8049 microcontroller to perform matrix scanning and interface communication between the keyboard and CPU.

Cable: 5-conductor, shielded cable. Attaches to a 5-pin DIN connector mounted on the computer backplane.

Refer to Figure A-18 in Appendix A for the schematic of the keyboard circuit.

The keyboard is a strobe scanning keyboard, controlled by an 8049 microcontroller. The microcontroller performs keyboard matrix scanning, setting and resetting of LEDs, actuation of the audio transducer, and serial communication with the system control processor on the I/O card.

All keys are configured in a general matrix consisting of 11 scan lines and 8 sense lines. A keyswitch and diode are arranged at each crossover point, to prevent generation of phantom keys due to multiple closures. Power (5 VDC) is applied to the keyboard through pin 5 of the keyboard cable. Refer to Table 15-1 for the device pinout of the 8049 keyboard microcontroller.

Table 15-1: Keyboard Processor Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
20	Vss	Circuit GND potential.
26	Vdd	Low power standby pin.
40	Vcc	Main power supply; + 5V during operation.
25	PROG	Output strobe for input/output expander.
27-34	P10 – P17 (Port 1)	8-bit quasi-bidirectional port.
21-24	P20 – 27 (Port 2)	Lower 4-bits of quasi-bidirectional port.
35-38		P20-P23 contain the four high-order program counter bits during an external program memory fetch and serve as a 4-bit input/output expander bus for the 8243.
12-19	D0 – D7	Bidirectional port that can be written or read synchronously using the $\overline{\text{RD}}$, $\overline{\text{WR}}$ strobes. The port also can be statically latched. Contains the 8 low-order program counter bits during an external program memory fetch, and receives the addressed instruction under the control of $\overline{\text{PSEN}}$. Also contains the address and data during an external RAM data store instruction under control of ALE, $\overline{\text{RD}}$, and $\overline{\text{WR}}$.
1	T0	Input pin testable using the conditional transfer instructions JT0 and JNT0. T0 can be designated as via clock output using ENT0 CLK instruction.
39	T1	Input pin testable using the JT1 and JNT1 instructions. Can be designated the timer/counter input using the STRT CNT instruction.
6	$\overline{\text{INT}}$	Interrupt input. Initiates an interrupt if interrupt is enabled. Interrupt is disabled after a reset. Also testable with conditional jump instruction (active low).
8	$\overline{\text{RD}}$	Output strobe activated during a BUS read. Can be used to enable data onto the bus from an external device. Used as a read strobe to external data memory (active low).

Table 15-1 (continued): Keyboard Processor Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
4	$\overline{\text{RESET}}$	Input used to initialize the processor (active low).
10	$\overline{\text{WR}}$	Output strobe during a bus write (active low). Used as write strobe to external data memory.
11	ALE	Address latch enable. This signal occurs once during each cycle. The negative edge of ALE strobes addresses into external data and program memory.
9	$\overline{\text{PSEN}}$	Program store enable. This output occurs only during a fetch to external memory.
5	$\overline{\text{SS}}$	Single-step input can be used in conjunction with ALE to "single-step" the processor through each instruction.
7	EA	External access input that forces all program memory fetches to reference external memory. Useful for emulation and debug, and essential for testing and program verification (active high).
2	XTAL1	One side of crystal input for internal oscillator.
3	XTAL2	Other side of crystal input.

Key Arrangement

Alphabetic Keys

Figure 15-1 identifies the 26 alphabetic keys, the CAPS LOCK, and the two SHIFT keys. The keys have the standard QWERTY typewriter arrangement. The two shift keys and CAPS LOCK key function in a manner similar to a typewriter. The exception is that the CAPS LOCK key affects only the alphabetic keys. All other keys are unaffected by its action.

If the small red light in the CAPS LOCK key is on, the condition is active and the alphabetic keys are entered as uppercase characters. If the light is off (inactive), the alphabetic keys are entered as lowercase characters unless one or both of the SHIFT keys are down at the time the key is pressed.

If the CAPS LOCK key is active (LED is on) and either of the shift keys are held down during an alphabetic key entry, the resulting characters will be defined as lowercase.

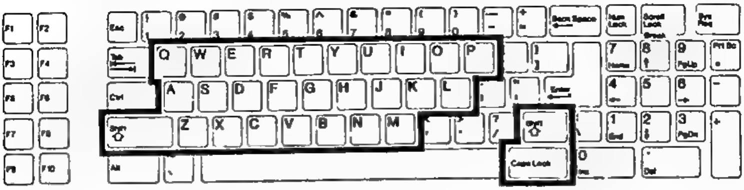


Figure 15-1: Alphabetic Keys

When a key is pressed, a key closure code representing the particular key is generated to the system control processor on the I/O card. The system control processor reads the code and then generates a new code referred to as a "system code." The operating system (and monitor program) use the system code to provide standard ASCII values for use by the application program.

When a key is released, a key opening code is generated. This code consists of the value 80H in combination with the value of the key's individual closure code. For example, when the A key is pressed, the closure 1CH is generated (then converted to system and ASCII codes). When the A key is released, 801CH is generated.

Table 15-2 details the key closure codes, system codes, and ASCII codes generated for the alphabetic keys. In addition, the table indicates the relative position (on the keyboard matrix) of each key (refer to Figure A-15 in Appendix A).

Table 15-2: Alphabetic Key Codes

	MATRIX	CLOSURE	SYSTEM	NOT		KEY PRESSED	
KEY	POSITION	CODE	CODE	SHIFTED	SHIFT	CTRL	ALT
A	30	1CH	1EH	1E61H	1E41H	1E01H	1E00H
B	48	32H	30H	3062H	3042H	3002H	3000H
C	46	21H	2EH	2E63H	3E43H	2E03H	2E00H
D	32	23H	20H	2064H	2044H	2004H	2000H
E	18	24H	12H	1265H	1245H	1205H	1200H
F	33	2BH	21H	2166H	2146H	2106H	2100H
G	34	34H	22H	2267H	2247H	2207H	2200H
H	35	33H	23H	2368H	2348H	2308H	2300H
I	23	43H	17H	1769H	1749H	1709H	1700H
J	36	3BH	24H	246AH	244AH	240AH	2400H
K	37	42H	25H	256BH	254BH	250BH	2500H
L	38	4BH	26H	266CH	264CH	260CH	2600H
M	50	3AH	32H	326DH	324DH	320DH	3200H
N	49	31H	31H	316EH	314EH	310EH	3100H
O	24	44H	18H	186FH	184FH	180FH	1800H
P	25	4DH	19H	1970H	1950H	1910H	1900H
Q	16	15H	10H	1071H	1051H	1011H	1000H
R	19	2DH	13H	1372H	1352H	1312H	1300H
S	31	1BH	1FH	1F73H	1F53H	1F13H	1F00H
T	20	2CH	14H	1474H	1454H	1414H	1400H
U	22	3CH	16H	1675H	1655H	1615H	1600H
V	47	2AH	2FH	2F76H	2F56H	2F16H	2F00H
W	17	1DH	11H	1177H	1157H	1117H	1100H
X	45	22H	2DH	2D78H	2D58H	2D18H	2D00H
Y	21	35H	15H	1579H	1559H	1519H	1500H
Z	44	1AH	2CH	2C7AH	2C5AH	2C1AH	2C00H

NOTE: The key opening code consists of the value 80H in combination with the value of the key closure code.

Nonalphabetic Keys

The nonalphabetic keys identified in Figure 15-2 include the numbers 0 through 9, the common punctuation marks, and the special programming characters that make up the remainder of the printable ASCII character set. Each of these keys can generate two characters. The uppercase character is generated when you press either SHIFT key in combination with another key.

Note that some keys do not produce a unique code when pressed with the CTRL or ALT key. In those instances, the code produced will be the same as if the CTRL or ALT key were not pressed and will be dependent upon the state of the SHIFT keys.

Table 15-3 details the key closure codes, system codes, and ASCII codes generated for the nonalphabetic keys. In addition, the table indicates the relative position (on the keyboard matrix) of each key (refer to Figure A-15 in Appendix A).

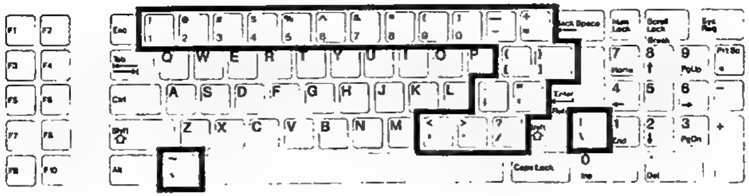


Figure 15-2: Nonalphabetic Keys

Table 15-3: Nonalphabetic Key Codes

KEY	MATRIX POSITION	CLOSURE CODE	SYSTEM CODE	NOT SHIFTED	SHIFT	KEY PRESSED CTRL	ALT
!/1	2	16H	02H	0231H	0221H		7800H
@/2	3	1EH	03H	0332H	0340H	0300H	7900H
#/3	4	26H	04H	0433H	0423H		7A00H
$/4	5	25H	05H	0534H	0524H		7B00H
%/5	6	2EH	06H	0635H	0625H		7C00H
↑/6	7	36H	07H	0736H	075EH	071EH	7D00H
&/7	8	3DH	08H	0837H	0826H		7E00H
*/8	9	3EH	09H	0938H	092AH		7F00H
(/9	10	46H	0AH	0A39H	0A28H		8000H
)/0	11	45H	0BH	0B30H	0B29H		8100H
_/–	12	4EH	0CH	0C2DH	0C5FH	0C1FH	8200H
+/=	13	55H	0DH	0D3DH	0D2BH		8300H
~/‘	43	0EH	29H	2960H	297EH		
{/[	26	54H	1AH	1A5BH	1A7BH	1A1BH	
}/]	27	5BH	1BH	1B5DH	1B7DH	1B1DH	
:/;	39	4CH	27H	273BH	273AH		
"/'	40	52H	28H	2827H	2822H		
</,	51	41H	33H	332CH	333CH		
>/.	52	49H	34H	342EH	343EH		
?//	53	4AH	35H	352FH	353FH		
|/\	55	5DH	2BH	2B5CH	2B7CH	2B1CH	

NOTE: The key opening code consists of the value 80H in combination with the value of the key closure code.

Common Control Keys

The common control keys are shown in Figure 15-3. Table 15-4 details the key closure codes, system codes, and ASCII codes generated for the common control keys. In addition, the table indicates the relative position (on the keyboard matrix) of each key (refer to Figure A-15 in Appendix A).

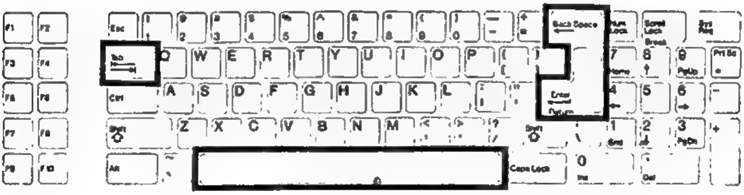


Figure 15-3: Common Control Keys

Table 15-4: Common Control Key Codes

	MATRIX	CLOSURE	SYSTEM	NOT		KEY PRESSED	
KEY	POSITION	CODE	CODE	SHIFTED	SHIFT	CTRL	ALT
BACK SPACE	14	66H	0EH	0E08H	0E08H	0E7FH	
TAB	15	0DH	0FH	0F09H	0F00H		
ENTER/ RETURN	28	5AH	1CH	1C0DH	1C0DH	1C0AH	
Space	57	29H	39H	3920H	3920H	3920H	3920H

NOTE: The key opening code consists of the value 80H in combination with the value of the key closure code.

The usual function of the BACK SPACE key moves the cursor one position to the left. This key is often used to erase the character preceding the cursor. The TAB key moves the cursor to the next tab column or to the previous tab column when used in combination with either SHIFT key. The ENTER/RETURN key moves the cursor to the left side of the display. Software usually adds a line feed when this occurs. The ENTER/RETURN key also is used to indicate completion of an operator entry. The space bar enters a blank character on the display.

Special Function Keys

The special function keys are identified in Figure 15-4. Table 15-5 details the key closure codes, system codes, and ASCII codes generated for the special function keys. In addition, the table indicates the relative position (on the keyboard matrix) of each key (refer to Figure A-15 in Appendix A).

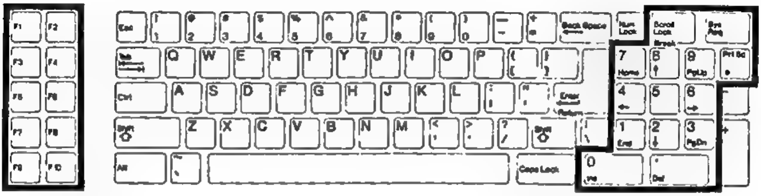


Figure 15-4: Special Function Keys

Table 15-5: Special Function Key Codes

	MATRIX	CLOSURE	SYSTEM	NOT		KEY PRESSED	
KEY	POSITION	CODE	CODE	SHIFTED	SHIFT	CTRL	ALT
Fl	59	05H	3BH	3B00H	5400H	5E00H	6800H
F2	60	06H	3CH	3C00H	5500H	5F00H	6900H
F3	61	04H	3DH	3D00H	5600H	6000H	6A00H
F4	62	0CH	3EH	3E00H	5700H	6100H	6B00H
F5	63	03H	3FH	3F00H	5800H	6200H	6C00H
F6	64	0BH	40H	4000H	5900H	6300H	6D00H
F7	65	02H	41H	4100H	5A00H	6400H	6E00H
F8	66	0AH	42H	4200H	5B00H	6500H	6F00H
F9	67	01H	43H	4300H	5C00H	6600H	7000H
F10	68	09H	44H	4400H	5D00H	6700H	7100H
SCROLL LOCK/ BREAK	69	7EH	46H	4600H			
SYS REQ	70	7FH	54H	5400H			
7 HOME	71	6CH	47H	4700H	4737H	7700H	7
8 ↑	72	75H	48H	4800H	4838H		8

Table 15-5 (continued): Special Function Key Codes

	MATRIX	CLOSURE	SYSTEM	NOT	KEY PRESSED		
KEY	POSITION	CODE	CODE	SHIFTED	SHIFT	CTRL	ALT
9 PGUP	73	7DH	49H	4900H	4939H	8400H	9
4 ←	75	6BH	4BH	4B00H	4B34H	7300H	4
5 %	76	73H	4CH	4C35H			5
6 →	77	74H	4D	4D00H	4D36H	7400H	6
PRT SC *	74	7CH	37H	372AH		7200H	
1 END	79	69H	4FH	4F00H	4F31H	7500H	1
2 ↓	80	72H	50H	5000H	5032H		2
3 PGDN	81	7AH	51H	5100H	5133H	7600H	3
0 INS	82	70H	52H	5200H	5230H		0
DEL .	83	71H	53H	5300H	532EH		

NOTE: The key opening code consists of the value 80H in combination with the value of the key closure code.

If the ALT key is used in combination with the BREAK (lowercase) key or the SYS REQ key, the keyboard buffer will be emptied. The CTRL key can be used in combination with the BREAK key or the SYS REQ key, to terminate an applications program. The SYS REQ key itself does not initiate a specific function unless directed to do so through software.

The HOME key will move the cursor to the home position on the screen (upper left-hand corner) and the PGUP key causes the cursor to move to the previous page of text (word processing).

When the PRT SC key is used in combination with either SHIFT key, the contents of the screen will be transferred to the printer port. This function is part of the monitor ROM and cannot be overridden. The * key (lowercase) enters an asterisk on the screen.

The END key will cause the cursor to be moved to the end of a document file (word processing) and the PGDN key causes the cursor to be moved to the next (following) page (word processing). The INS key causes the applications program (word processing) to enter the insert mode. The DEL key is used to delete characters (word processing).

The 0-9 keys enter the respective digit (for each key) when the NUM LOCK key is active (LED on). The cursor control keys allow the user to move the cursor to any desired position on the screen.

Control Keys

The control keys are identified in Figure 15-5. The ALT and CTRL keys do not generate a code but modify the codes of the other keys.

The ESC key is located at matrix position 1 (Figure A-15 in Appendix A) and generates a closure code of 76H, a system code of 01H and an ASCII code of 1BH (011BH). The ESC function is often used by software to stop the execution of a program or, when used in sequence with another key, as a means of entering escape codes required by some software.

The ALT key can be used to generate any hexadecimal code from 0 to FFH (0 to 255 decimal). To do so, press and hold the ALT key and then enter the decimal equivalent of the hexadecimal code you wish to generate. Then release the ALT key. When the ALT key is released, the conversion takes place and the hexadecimal code is generated.

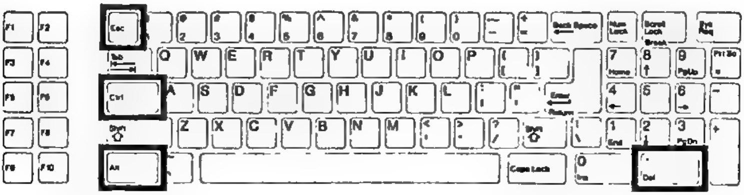


Figure 15-5: Control Keys

Special Keys

Figure 15-6 identifies the special keys that are used with the numeric keypad.

The NUM LCK key is used to "lock" the keypad into the shifted position. When locked, the SHIFT key produces the "not shifted" codes in Table 15-5 for the 0 through the 9 and "." keys of the keypad. This allows the operator to use the keypad in a manner similar to a 10-key calculator. However, through software, it would be possible to use the keypad for other purposes.

The minus key (−) is located at matrix position 78 (Figure A-15 in Appendix A) and generates a closure code of 7BH, a system code of 4AH and an ASCII code of 2DH (4A2DH). The plus key (+) is located at matrix position 84 (Figure A-15 in Appendix A) and generates a closure code of 79H, a system code of 4EH and an ASCII code of 2BH (4A2BH). The ENTER/RETURN key is located at matrix position 28 (Figure A-15 in Appendix A) and generates a closure code of 5AH, a system code of 1CH and ASCII codes 540DH (shifted and unshifted modes) and 540AH (in combination with the CTRL key).

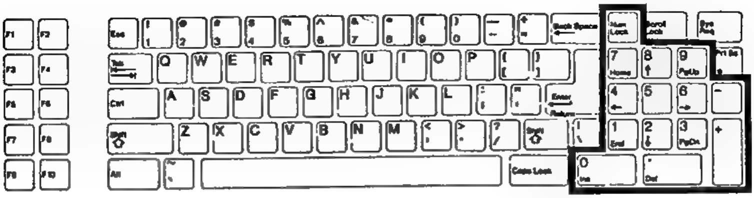


Figure 15-6: Keypad Special Keys

Increasing Keyboard Buffer Size

You may increase the keyboard buffer size if you find 15 characters are insufficient for your applications. To set a buffer size to some value other than these 15 characters, first determine the size of the buffer that you wish to establish, then use the following procedure:

- Set the word variable at F000:C8 to the segment address for this block of memory.
- Set the word variable at 40:1A, 40:1C, and F000:CA to the starting offset of the new keyboard buffer.
- Set the word variable at F000:CC to the offset of the last byte in the keyboard buffer.

NOTE: You cannot change the buffer size through the commands in the monitor ROM. Several of the instructions must be carried out before any input from the keyboard will be properly recognized. Therefore, you must use an assembler, such as the Microsoft Macro Assembler, available as part of the Programmer's Utility Pack from Zenith Data Systems.

The following assembly language example sets aside the top 8K of unused monitor RAM as a 4K character keyboard buffer.

```
MONITOR_SEGMENT SEGMENT AT (0F000H)                 ;Monitor data segment
KEY_BUFF_SEGMENT            EQU  0C8H               ;Buffer segment
KEY_BUFF_START              EQU  0CAH               ;Start offset of buffer
KEY_BUFF_END                EQU  0CCH               ;End offset of buffer
FREE_MEM                    EQU  2000H              ;Start of available memory
END_MEM                     EQU  3FFFH              ;End of free memory
MONITOR_SEGMENT ENDS

DATA_SEGMENT          SEGMENT  AT   (40H)           ;Compatible data segment
KEY_HEAD_PTR                EQU  1AH                ;Buffer head pointer
KEY_TAIL_PTR                EQU  1CH                ;Buffer tail pointer
DATA_SEGMENT ENDS

CODE  SEGMENT PUBLIC

      ASSUME    CS:CODE,          DS:DATA_SEGMENT,     ES:MONITOR_SEGMENT

           ORG  100H                                ;Start of .COMprogram
BGKBF:     MOV  AX,MONITOR_SEGMENT                  ;Point to monitor segment
           MOV  ES,AX
           MOV  AX,DATA_SEGMENT                     ;Point to data segment
           MOV  DS,AX
           MOV  ES:KEY_BUFF_SEGMENT,MONITOR_SEGMENT ;Set buffer segment
           MOV  DS:KEY_HEAD_PTR,FREE_MEM            ;Set buffer head ptr
           MOV  DS:KEY_TAIL_PTR,FREE_MEM            ;Set buffer tail ptr
           MOV  ES:KEY_BUFF_START_,FREE_MEM         ;Set start of buffer
           MOV  ES:KEY_BUFF_END,END_MEM             ;Set end of buffer
           RET                                      ;Done! - exit.

CODE ENDS

           END BGKBF
```

To compile the above code using the macro assembler, enter the following commands:

```
MASM BIGKBUFF;
LINK BIGKBUFF;
EXE2BIN BIGKBUFF.EXE .COM
```

The MASM file is only on the Programmer's Utility Pack distribution disk. The LINK and EXE2BIN files are both on the second distribution disk of MS-DOS. You should not see any errors in the MASM sequence or the EXE2BIN operation. You will see an error in the link sequence (missing stack segment), which is normal.

Appendices

The appendices in this part of the manual supply technical information for those assemblies that are not considered part of the basic Z-248 computer. Each appendix also includes complete operating specifications and configuration information for the given assembly.

If you install optional equipment in your computer, you may also install applicable appendices in this manual to form a convenient extension of the system hardware section. Each technical manual can be individually customized to represent the specific hardware configuration of a particular computer system.

Notes

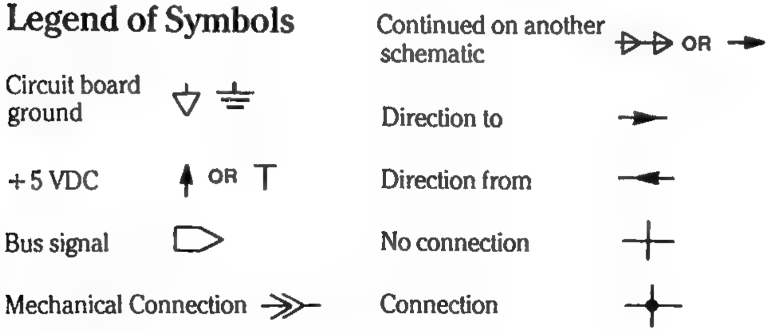
1. The symbols on the schematic illustrations in this appendix represent the logic flow of the circuit rather than any device's specific design.

2. Active low signals may be indicated by overscoring, the tilde (~), the asterisk (*), or the backslash (\) characters used in combination with a signal name.

3. All resistor values are in ohms (K = 1,000, M = 1,000,000). All resistors are 1/4 watt, 5% tolerance unless otherwise specified.

4. All capacitor values are in μF (microfarads) unless otherwise specified.

Legend of Symbols



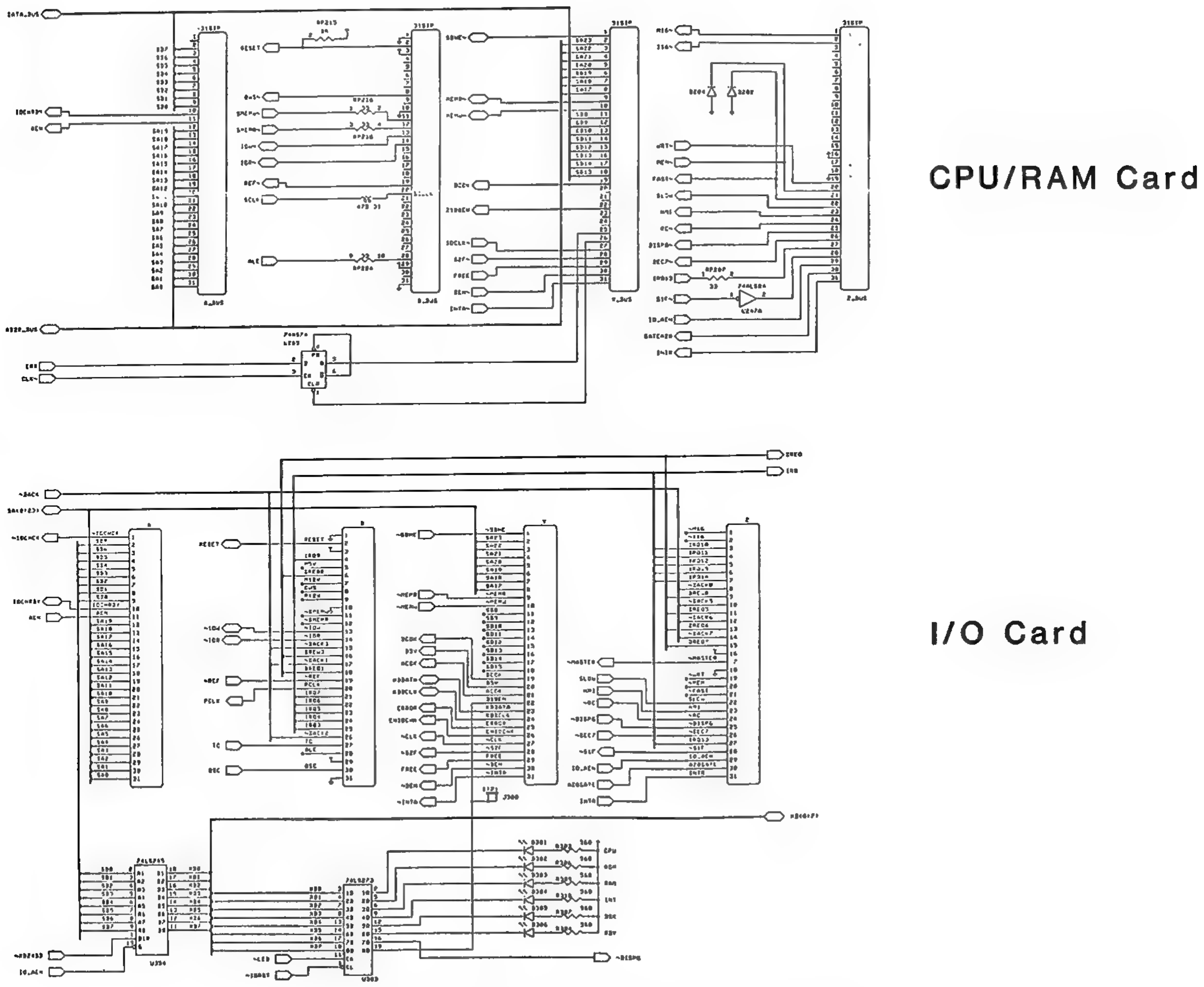


Figure A-1: Backplane Signal Interface Schematic

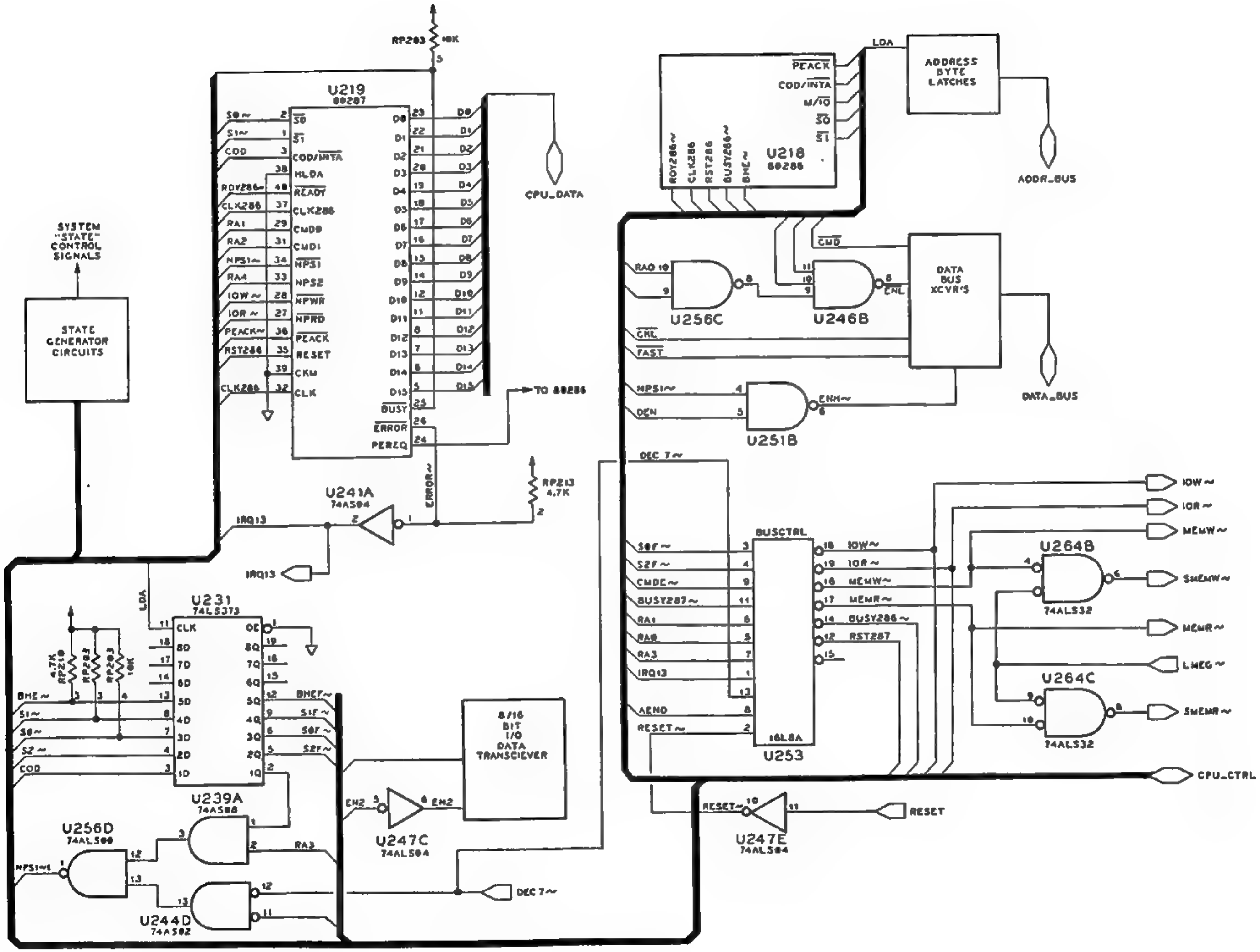


Figure A-2: Control Bus Schematic

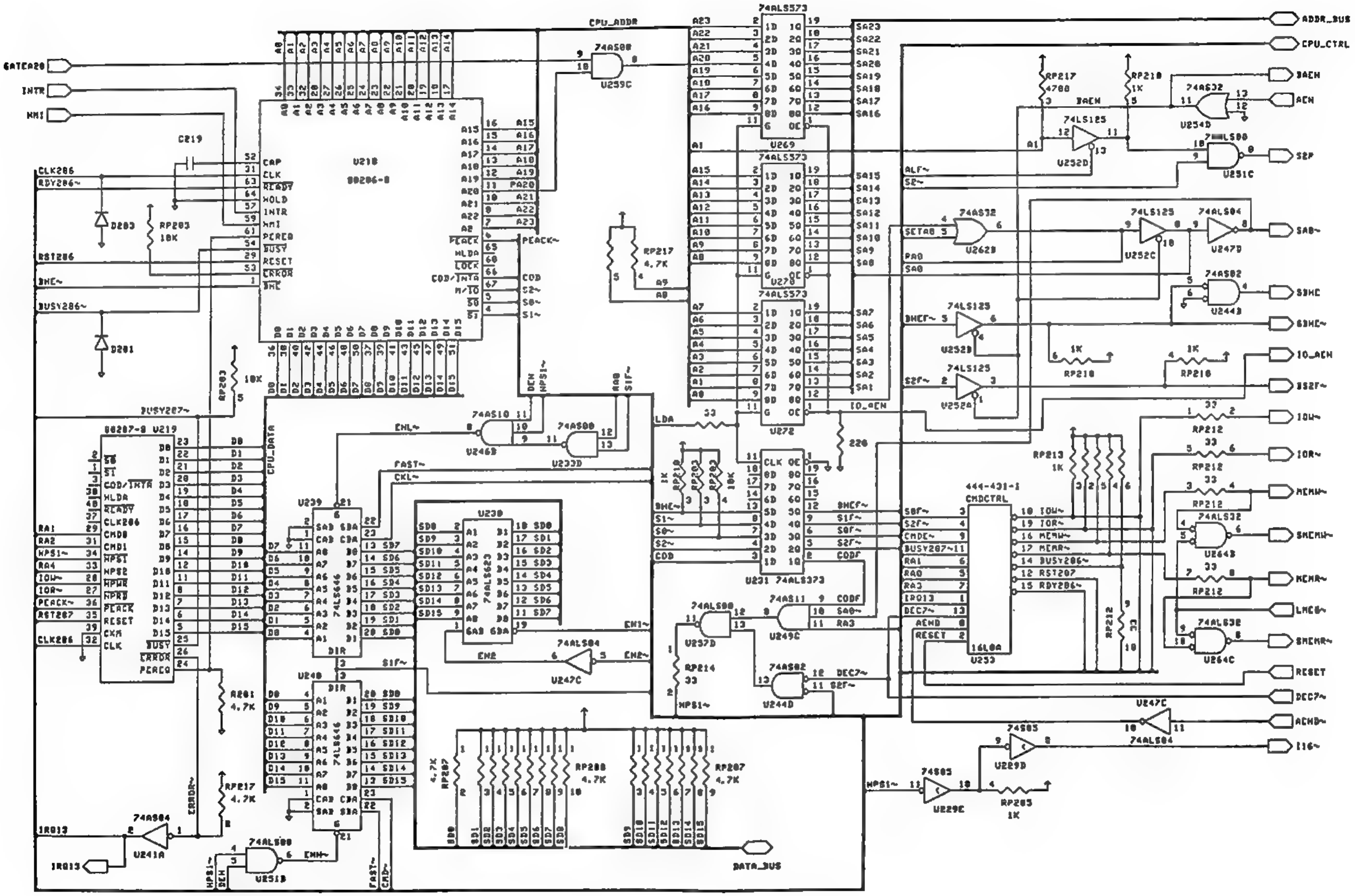


Figure A-3: CPU Control Logic Schematic, Part 1

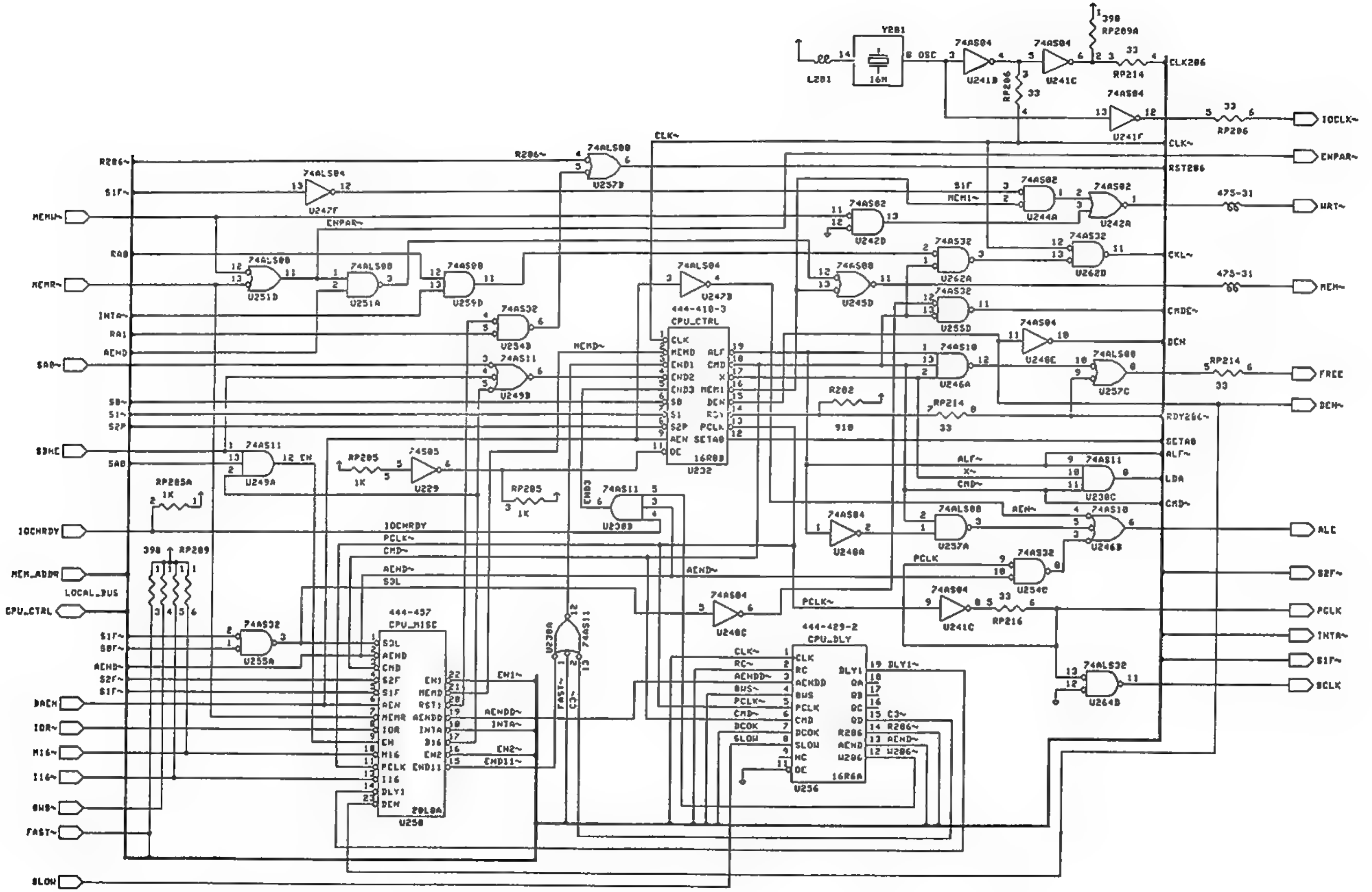


Figure A-4: CPU Control Logic Schematic, Part 2

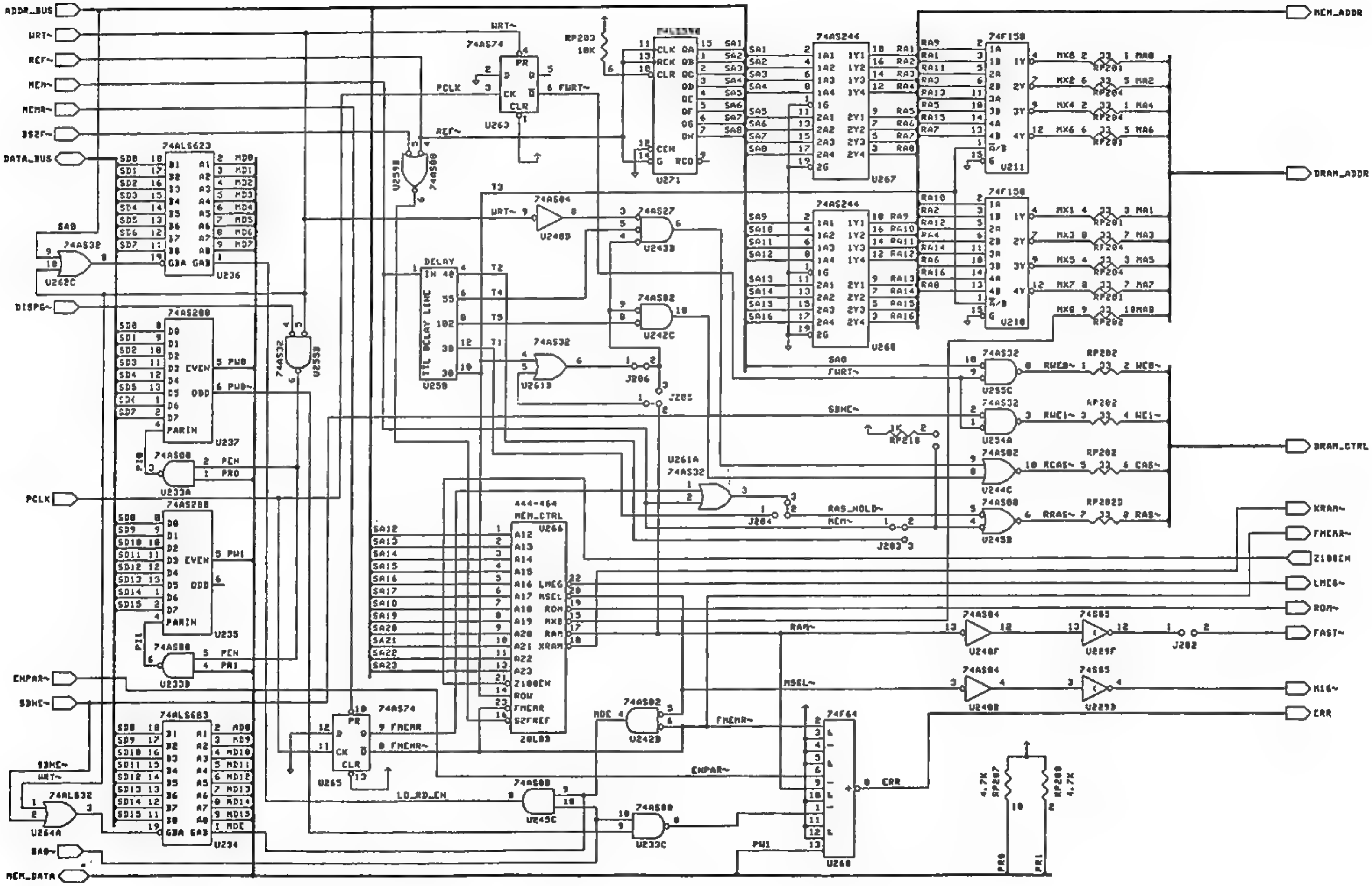


Figure A-5: Memory Control Logic Schematic

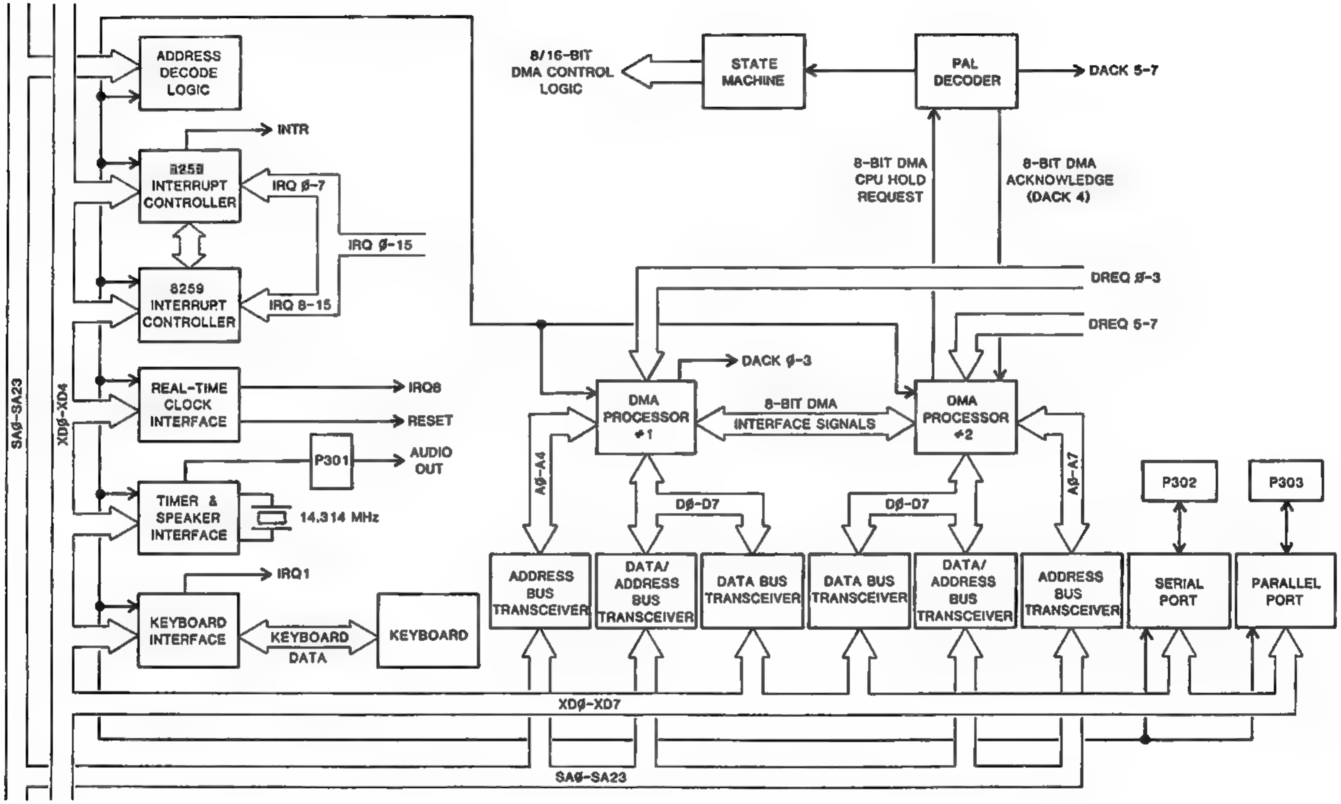


Figure A-6: I/O Card Block Diagram

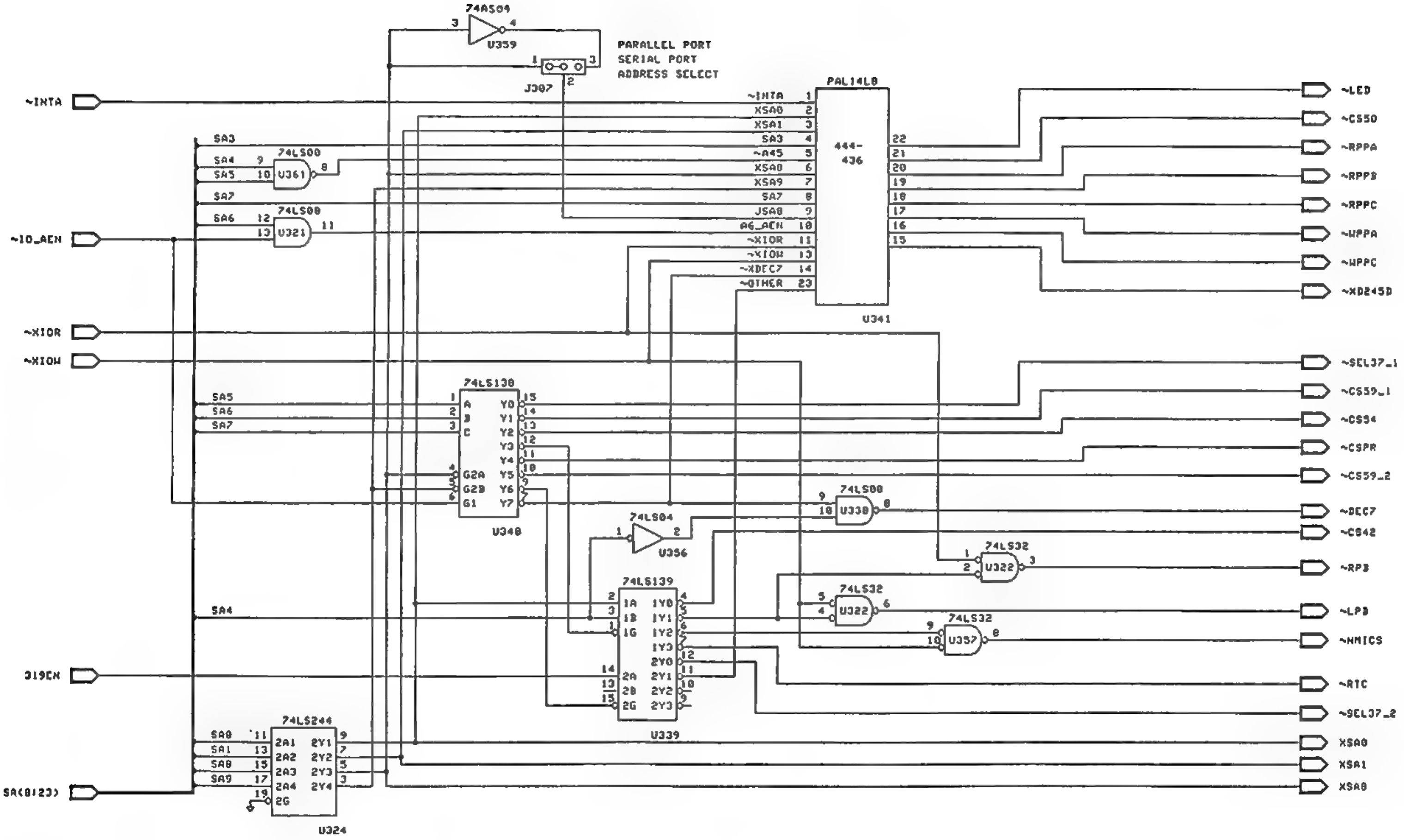


Figure A-7: Address Decode Logic Schematic

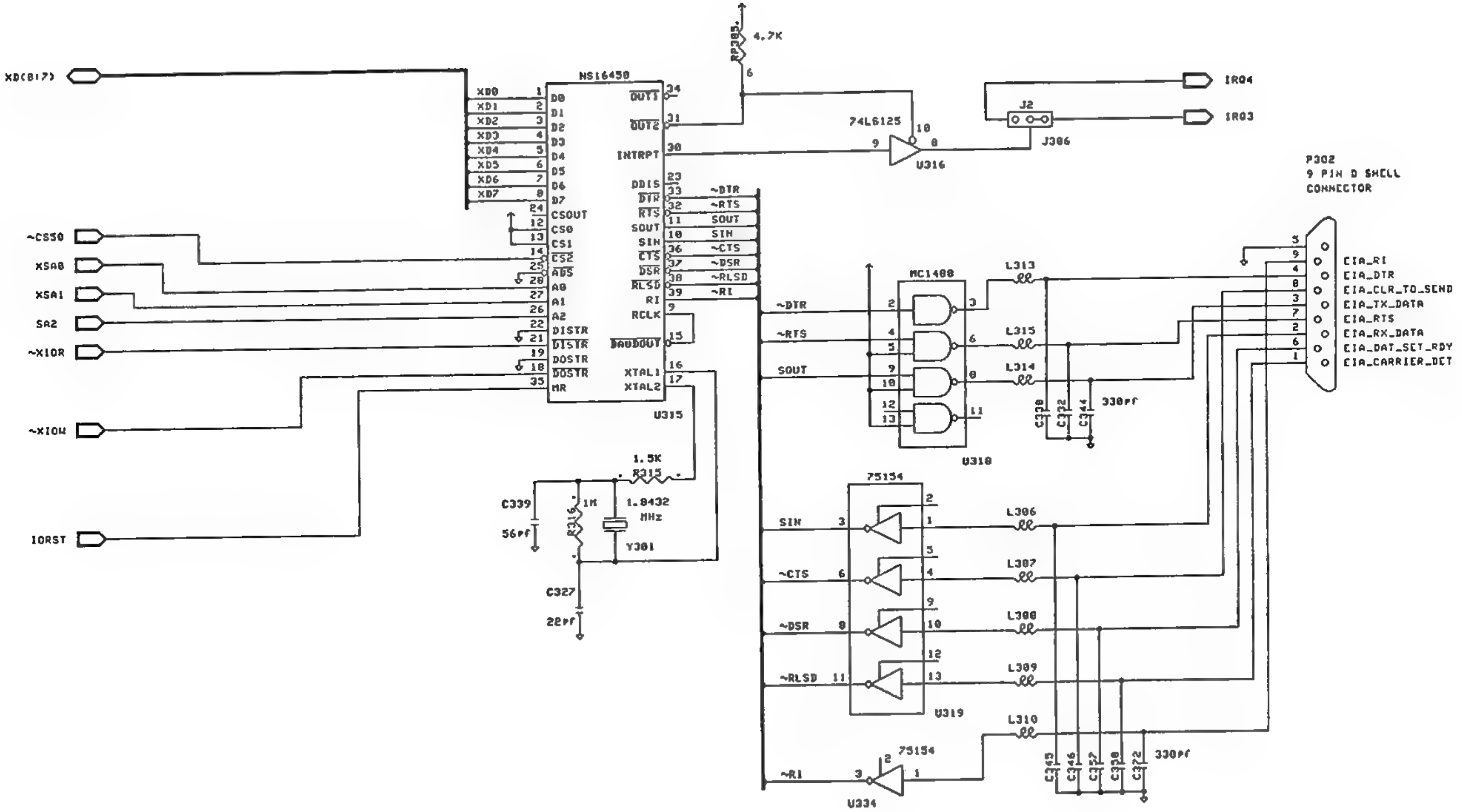


Figure A-8: Serial Port Schematic

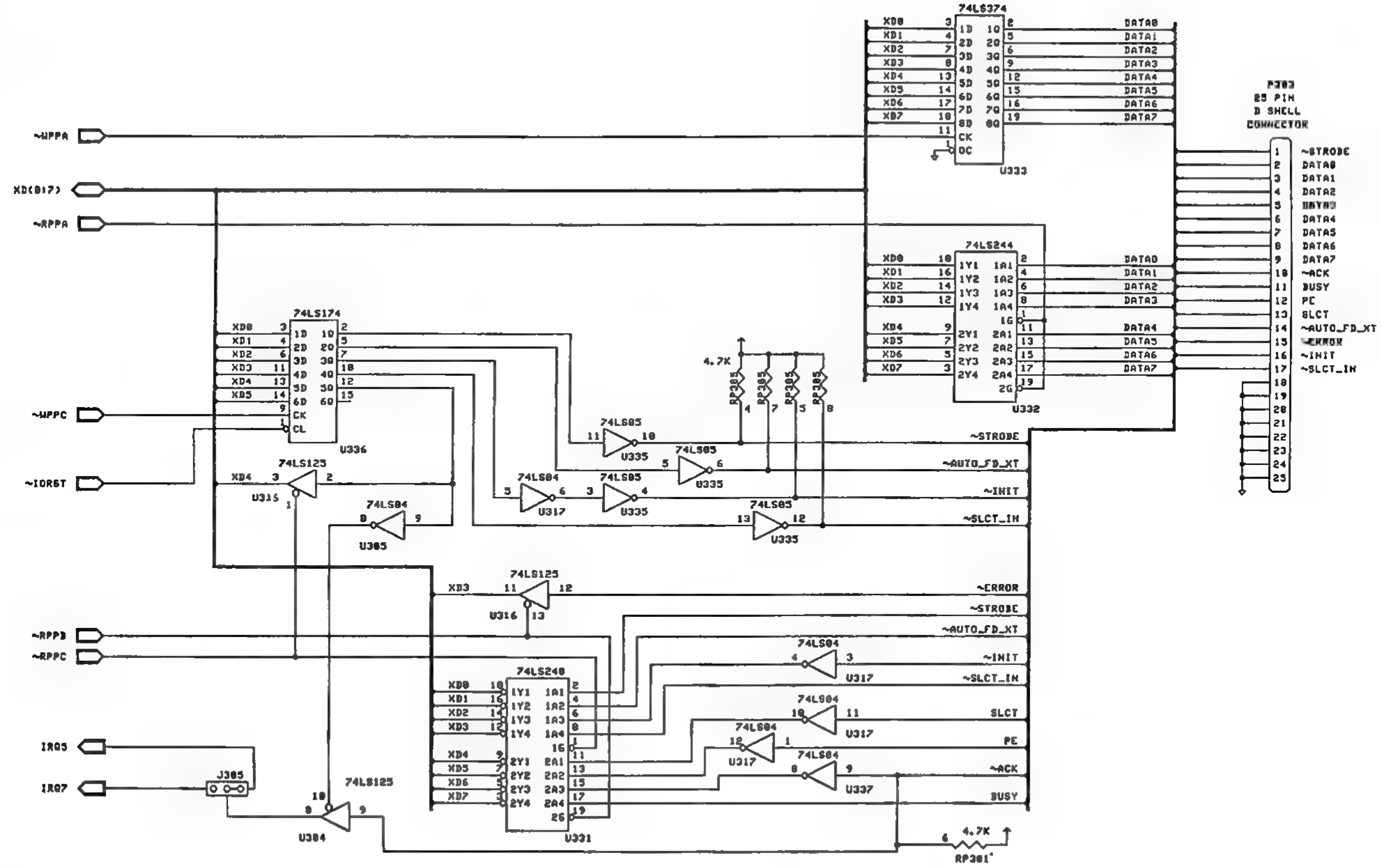


Figure A-9: Parallel Port Schematic

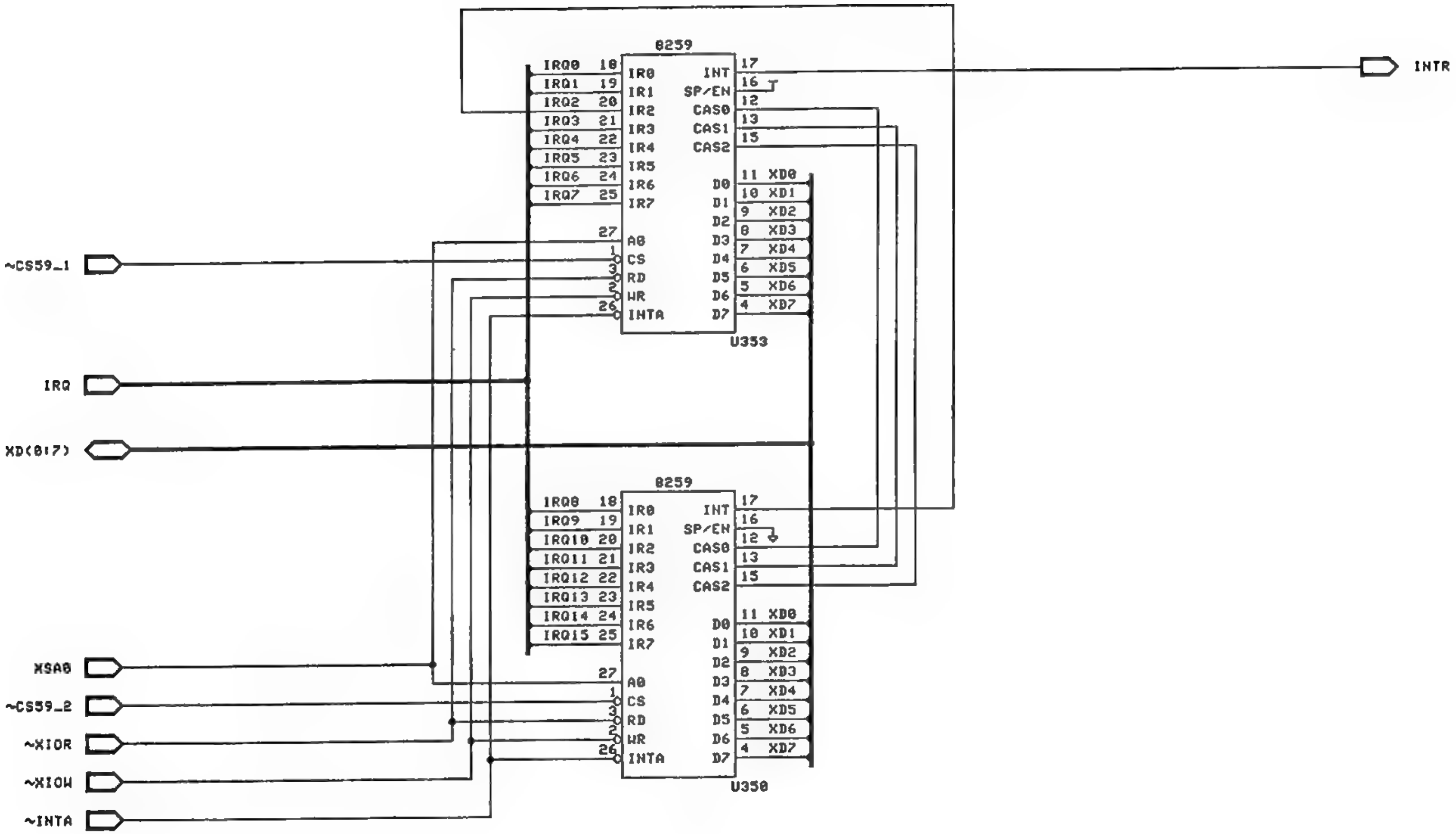


Figure A-10: Interrupt Control Schematic

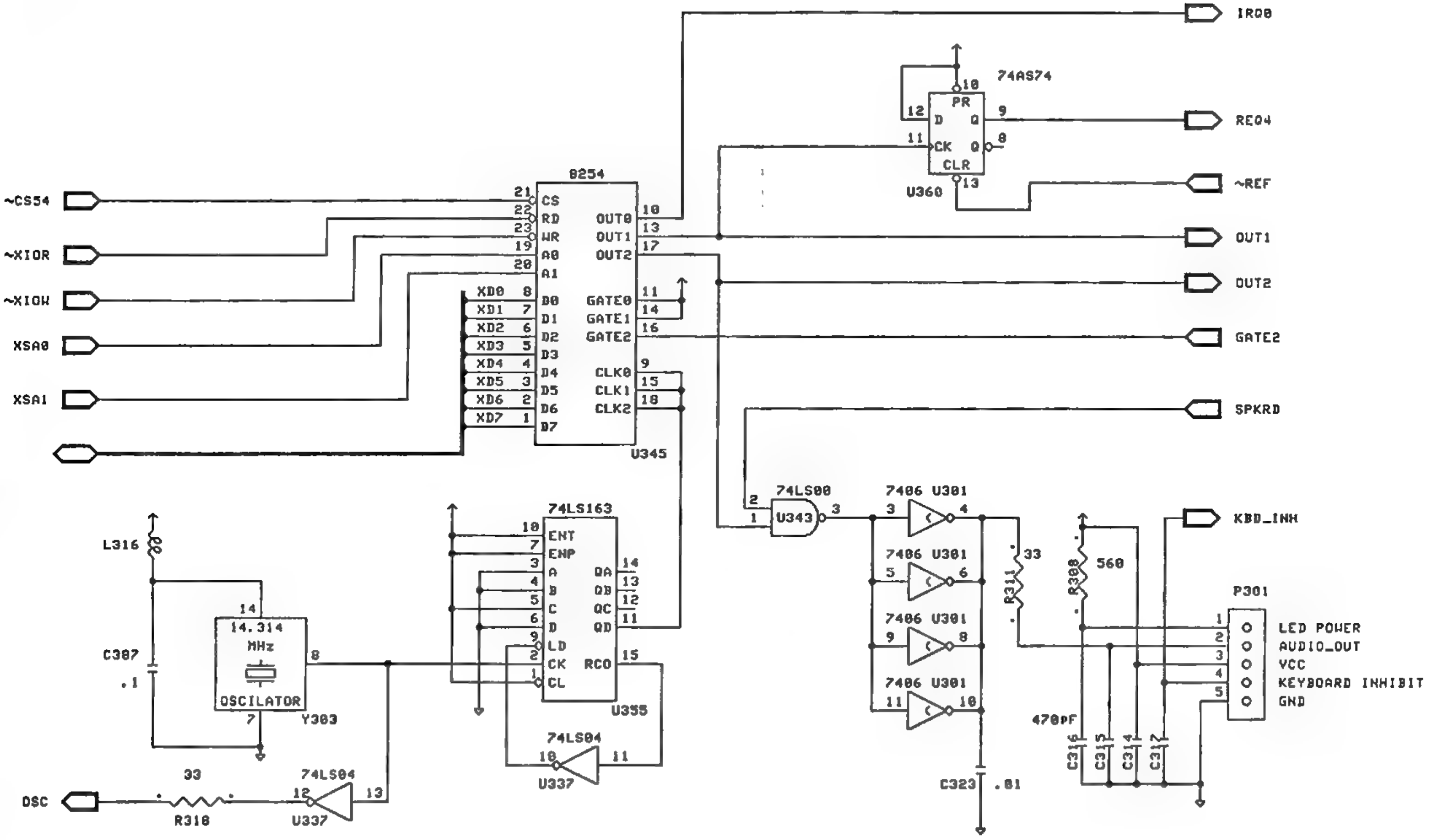


Figure A-11: Programmable Interval Timer Schematic

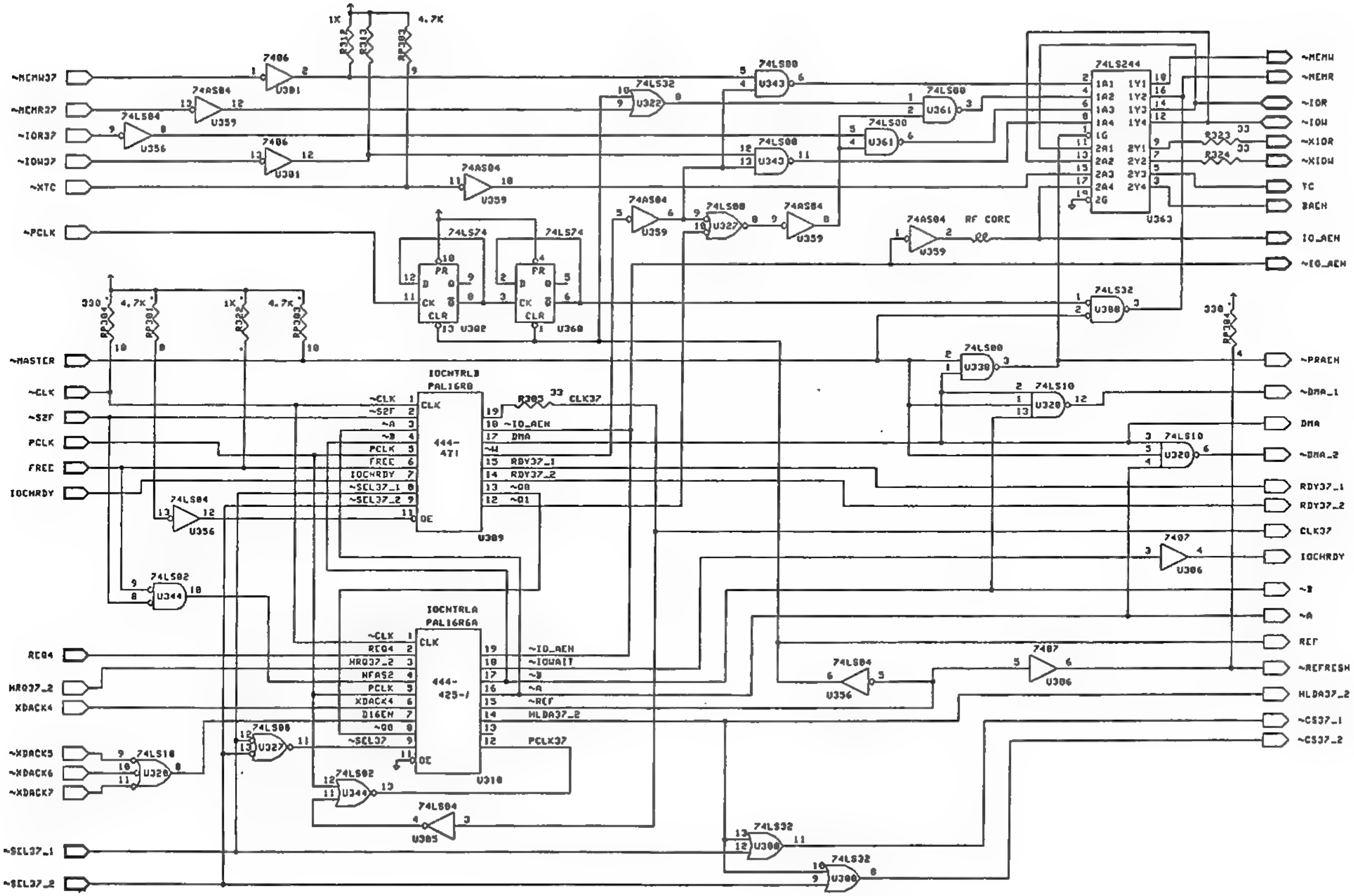


Figure A-12: DMA Control Schematic, Part 1

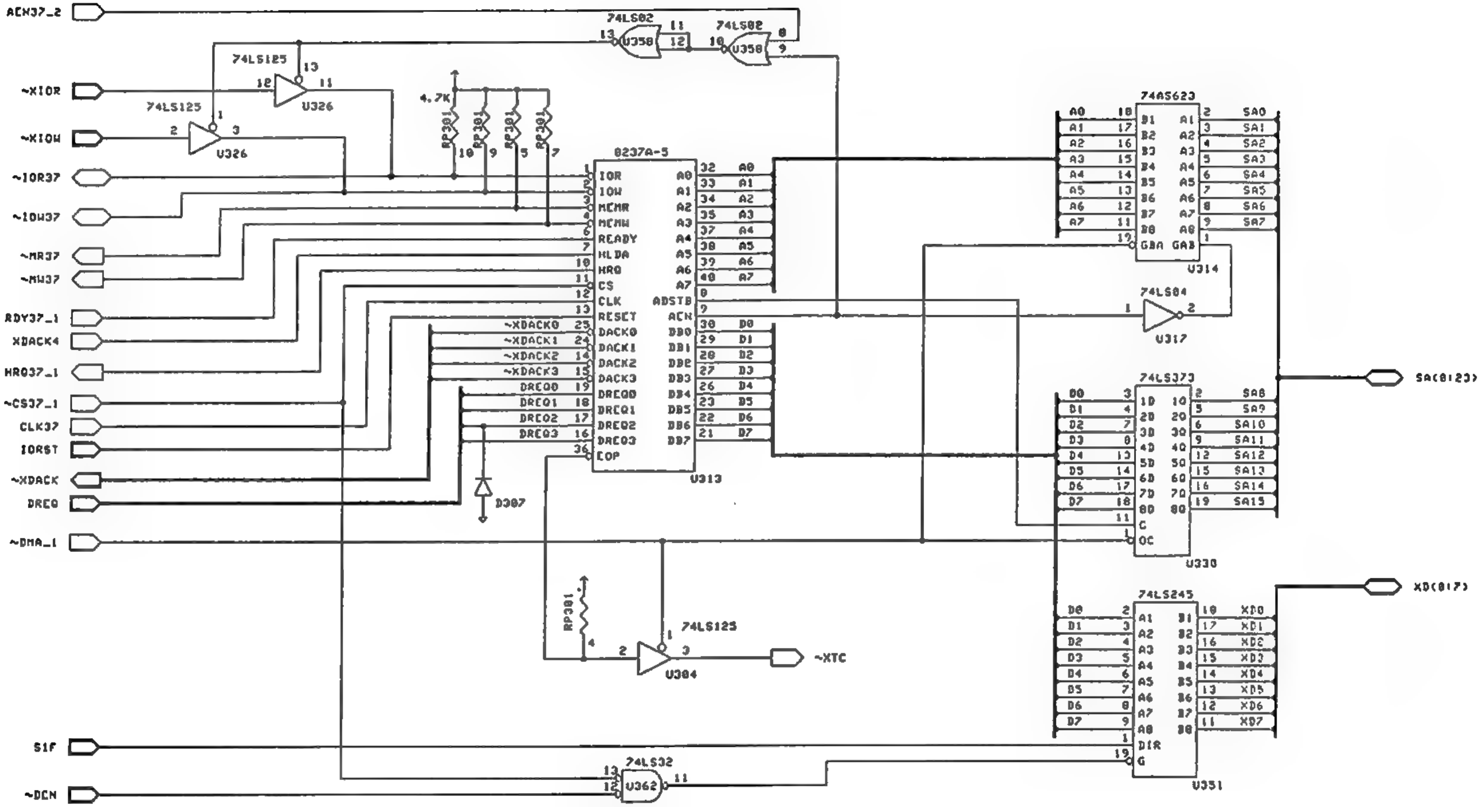


Figure A-13: DMA Control Schematic, Part 2

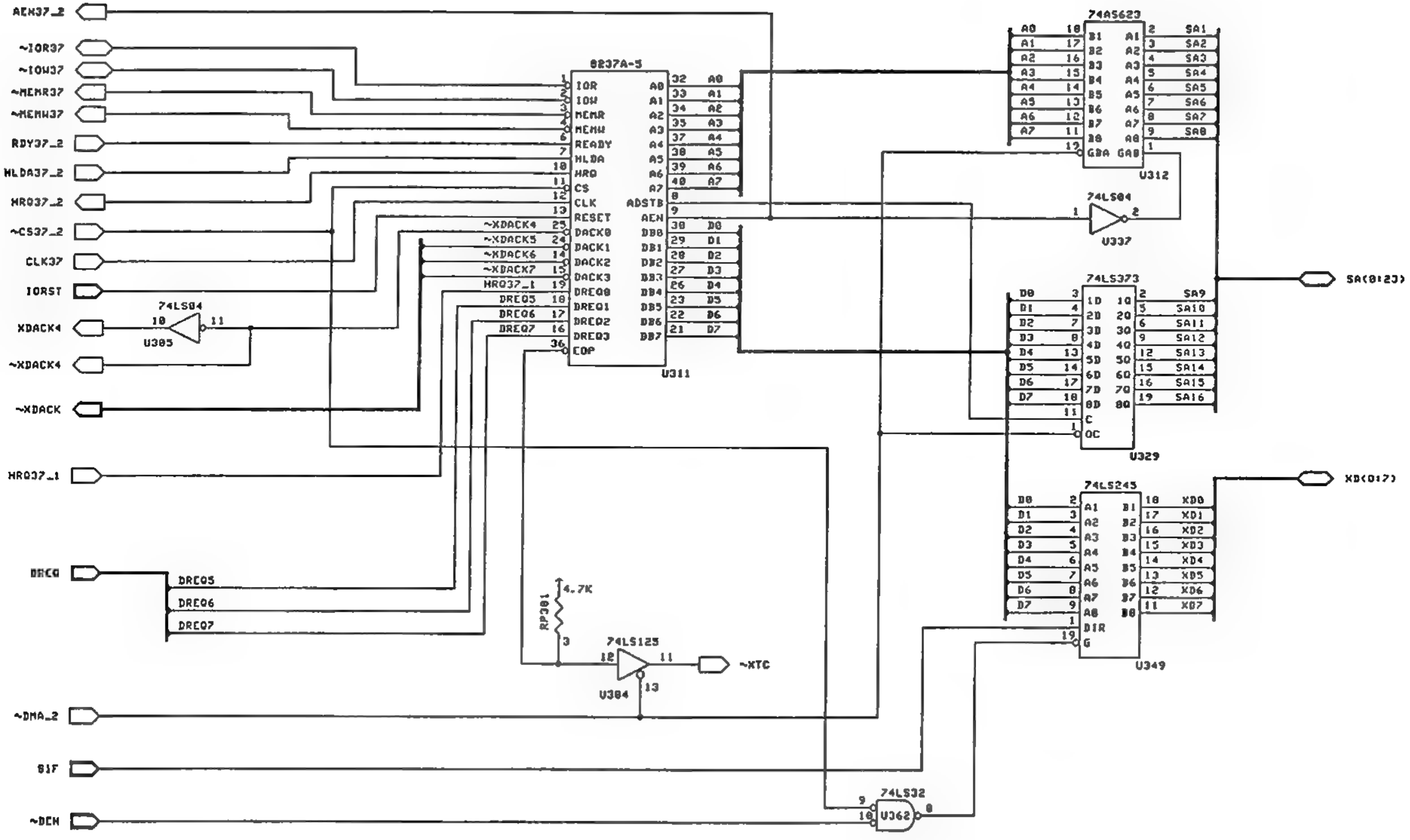


Figure A-14: DMA Control Schematic, Part 3

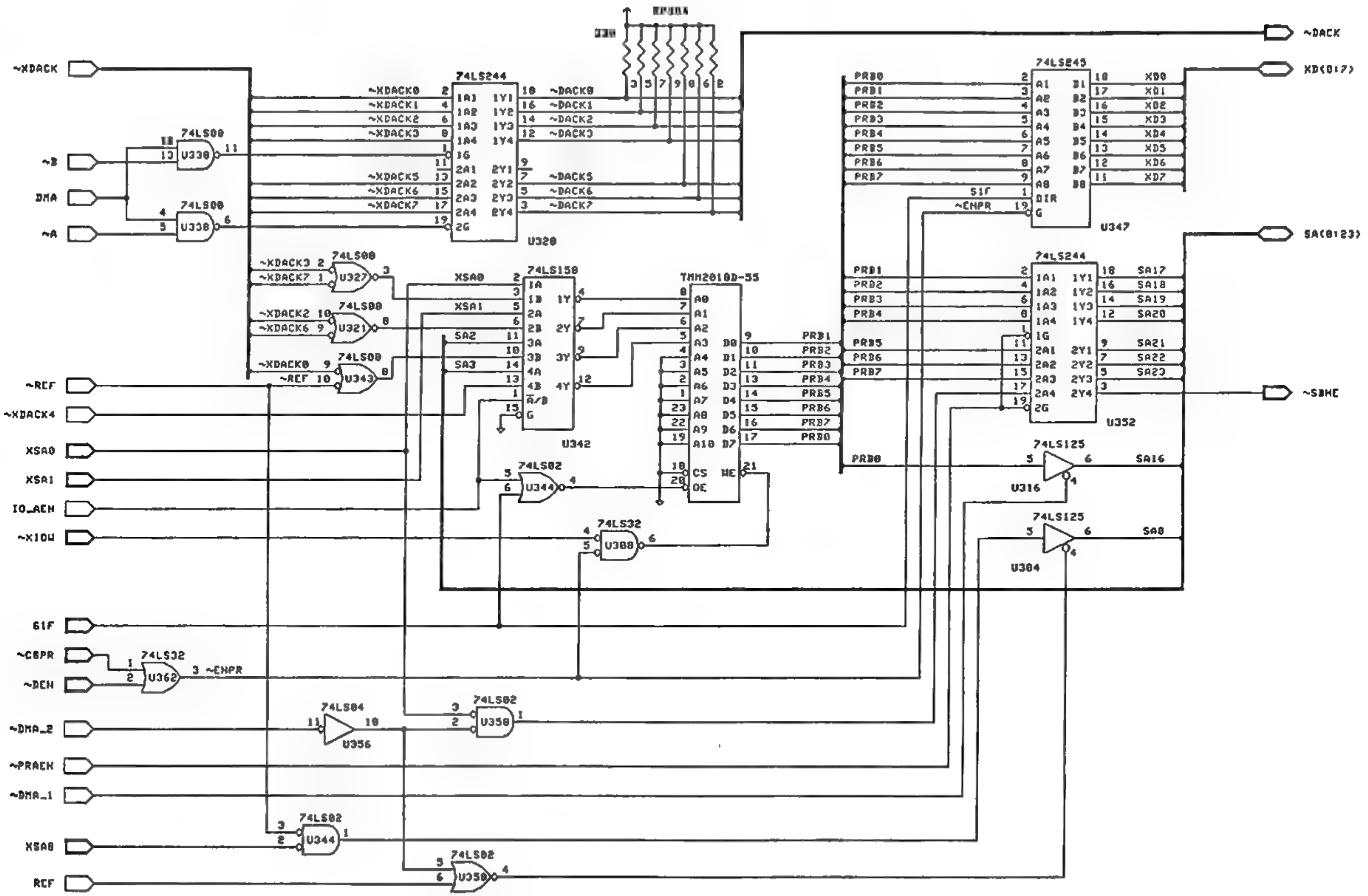


Figure A-15: DMA Control Schematic, Part 4

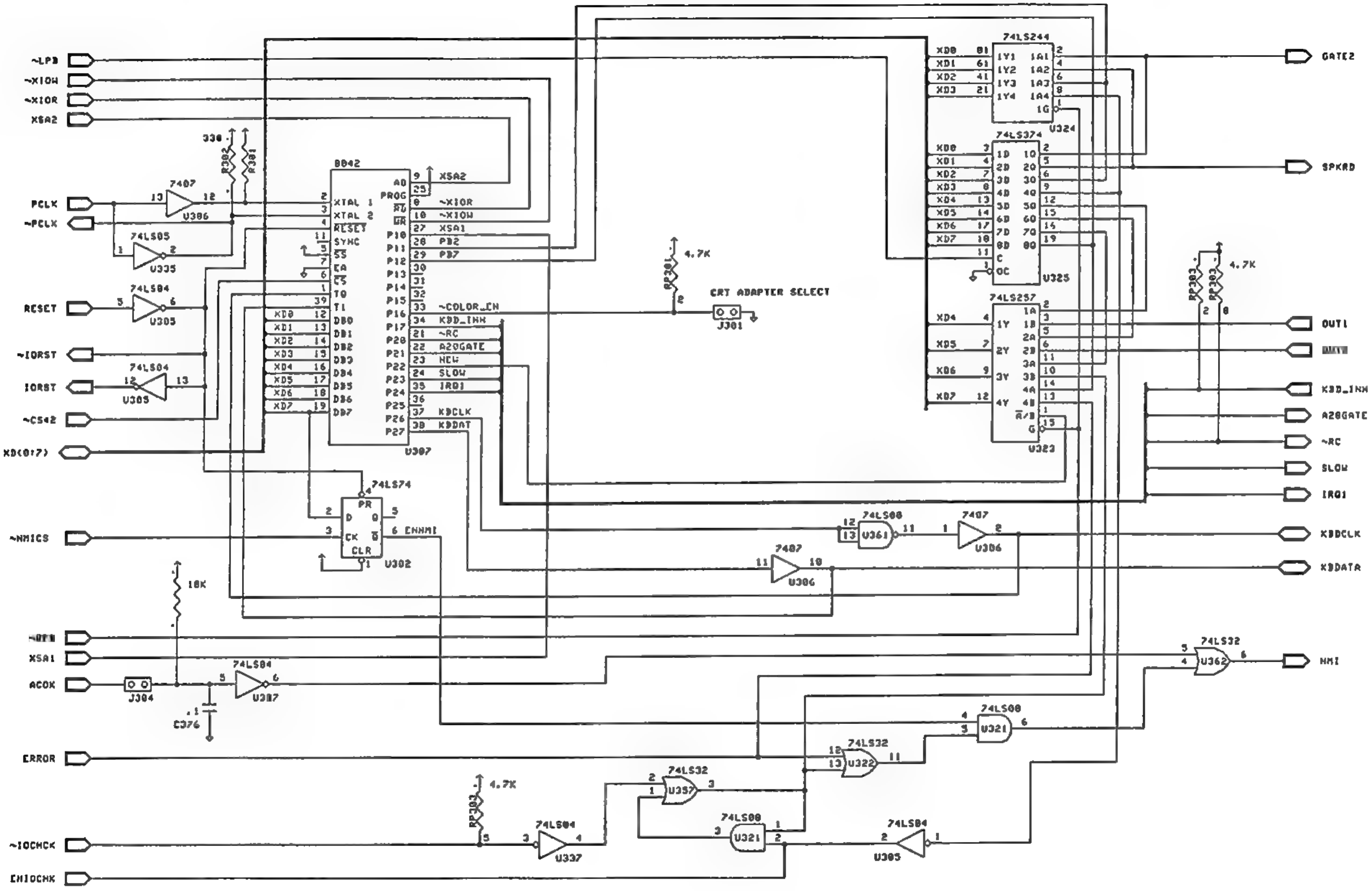


Figure A-16: Keyboard Interface Schematic

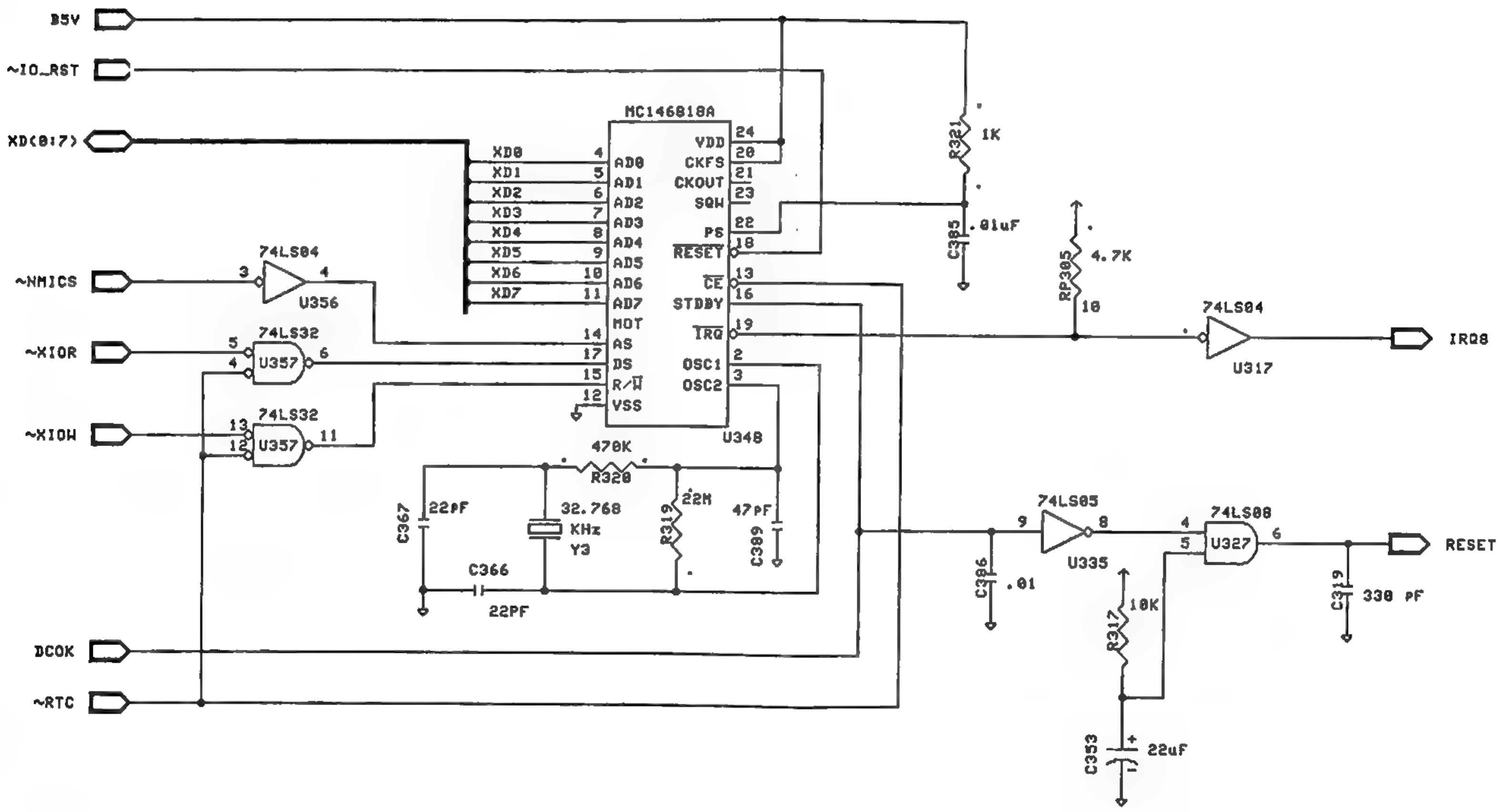


Figure A-17: Real-Time Clock Interface Schematic

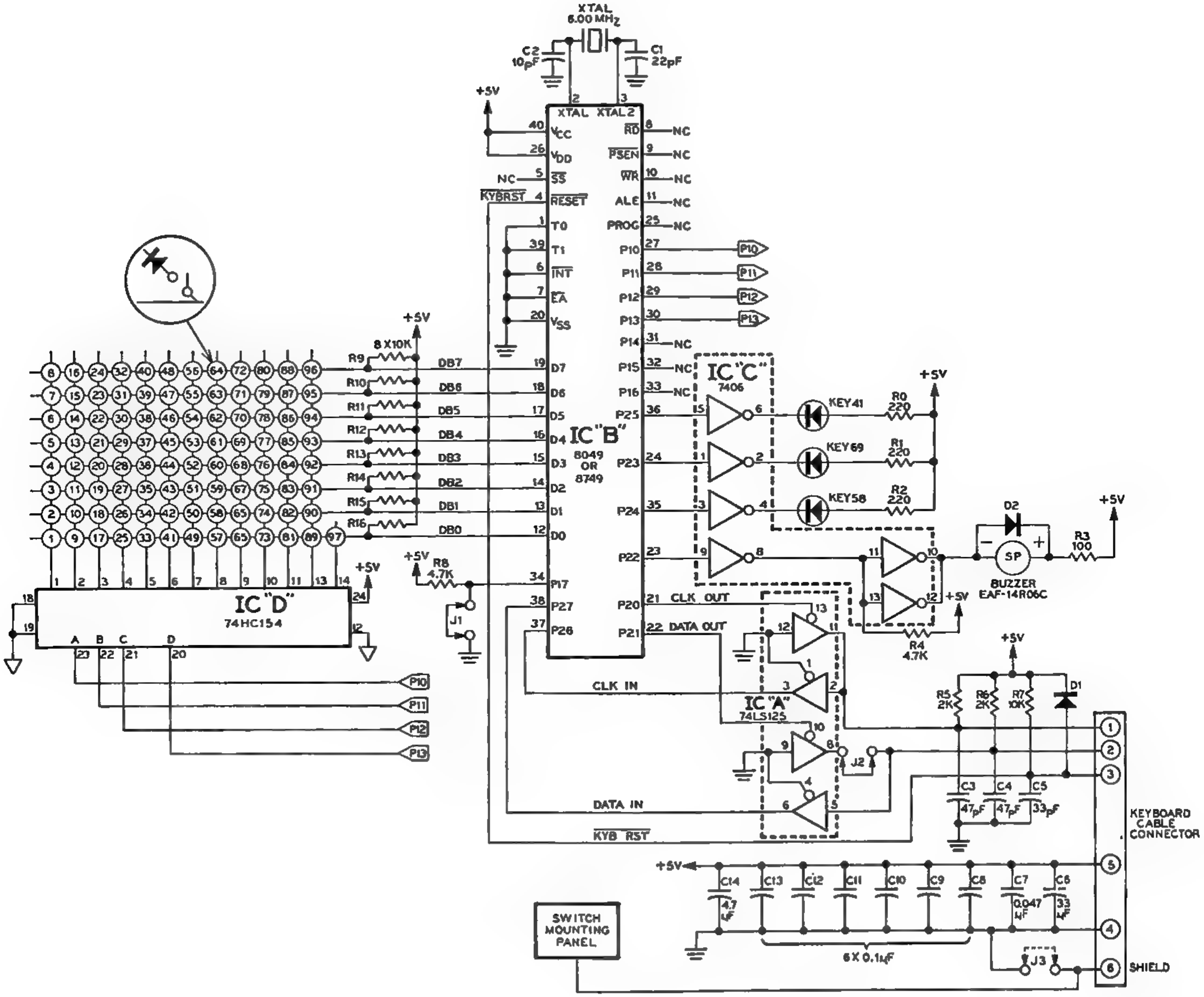


Figure A-18: Keyboard Schematic

Index

1
1.2M floppy disk drives, see Disk drives

3
360K floppy disk drives, see Disk drives

6
6845 CRT Controller, 2-19

8
80286 Instruction cycle, 13-7
80286 Microprocessor, 1-2, 2-4, 13-2, 13-6, 14-21
        see also CPU/RAM Card
80287 Numeric processor extension, 1-2, 13-4, 13-6, 13-16
8042 Universal peripheral interface, 14-21
8049 Microcontroller, see Keyboard
8259 Programmable interrupt controller, 6-10, 7-3, 14-10
    Initialization, 14-11

A
Address
    highest and lowest segment, 7-15
    memory, 7-15
    offset, 7-15
    pointing to memory, 7-15
    segment, 7-15
Address bus, 2-4, 13-2
Address mode
    protected, 13-2
    real, 13-2
Alphabetic keys, 15-4
Appendix A, 16-1
    legend of schematic symbols, 16-1
    schematic illustrations, 16-1
    schematic notes, 16-1
Arithmetic overflow, 7-2
ASCII codes, see Keyboard
Audio, 14-15
Autoboot, 3-1, 3-3

B
Backplane, 12-6
battery backup circuit, 1-2
edge connectors, 2-2
keyboard receptacle, 1-2
LEDs, see Diagnostic LEDs
overview, 2-2
PC AT compatible edge connectors, 1-2, 12-6
PC-compatible edge connectors, 1-2, 12-6
specifications, see Specifications
Zenith edge connectors, 1-2, 12-6
Base memory, see Setup program
Basic computer, 1-2
Battery
backup, see Lithium battery
lithium, see Lithium battery
Bit cell space, 2-9
Bit shift, 2-14
Boot, 3-2, 3-3, 5-5
Boot command, 5-5
Boot drive, see Setup program
Boot, 3-1
see also Autoboot
Booting, 10-11
Bus
address, 2-4, 13-2
control, 2-4, 13-2, 13-6
data, 2-4, 13-4
memory address, 13-20, 13-22
memory control, 13-24
Bus cycle control logic, 13-9
Bus description, 12-8
PC AT Compatible, 12-11
PC-Compatible, 12-8
Zenith Data Systems, 12-13

C
Cable
floppy disk drive interface, 5-14, 5-20
keyboard, see Keyboard
Winchester disk drive interface, 5-22
Calculator keypad, see Keyboard

Card
  CPU/RAM, see CPU/RAM card
  data storage controller, see Options
  I/O, see I/O card
Central prrocessing unit, 13-2
Character ROM, see Optional video interface
Clock, 13-17
  real-time, see Real-time clock
Common control keys, see Keyboard, 15-8
Configuration jumpers, see Hardware jumpers
    see also Setup program
    see also Slow mode
    see also Z-100 mode
Connections
  keyboard, 2-22
  parallel port, 14-7
  serial port, 14-4
Control bus, 2-4, 13-2, 13-6
Control Keys, see Keyboard
Control logic, see CPU
Controller
  6845 CRT Controller, 2-19
  DMA, 2-5, 14-2, 14-15
  keyboard, 2-22
  NS16450 Programmable asynchronous communications, 14-4
  programmable interrupt, 14-2, 14-10
Controlling timing loops, 7-12
Cover replacement, 4-10
CPU
  control logic, 13-14
CPU architecture, see Pipeline architecture
CPU, see Central processing unit
CPU/RAM card, 1-3, 2-4, 13-1
  80286 Microprocessor, 13-2
  80287 Numeric processor extension, 13-16
  block diagram, 13-1
  central processing unit, 13-2
  CPU wait states, 1-3, 5-8, 13-15
  parity, 1-3
  RAM (Random access memory), 1-3
  random access memory, 13-20
  ROM (Read-only memory), 1-3

CRT retrace, 2-21
CRT scanning, 2-21
CRT signal relationships, 2-21
CRT/Monitor operation, 2-20
Cursor control keys, 5-1

D
Data bus, 2-4, 13-4
Data encoding
  FM, 2-8, 2-16
  MFM, 2-9
  MMFM, 2-10
Data Storage Controller Card, see Options
Data storage devices, 2-15
  floppy disk drives, 2-15
  Winchester disk drives, 2-17
Data transfer rates, 2-15
Data transmission, 2-6
Date, see Setup program
Daylight Savings Time, see Setup program
Device control, 7-5
Diagnostic LEDs
  backplane, 1-2, 2-3, 12-8
  I/O Card, 2-6
Dipoles, 2-8, 2-13
Disassembly, 4-1
  backplane board removal, 4-4
  circuit card removal, 4-2
  cover removal, 4-1
  floppy disk drive removal, 4-7
  power supply removal, 4-6
  tape backup unit, 4-10
  Winchester disk drive removal, 4-9
Disk drives, 1-3, 2-16
  1.2M floppy, 1-3, 2-15
  360K floppy, 1-3, 2-15
  bytes per sector, 10-5
  controller option byte, 10-6
  disk side, 10-5
  drive function calls, 10-3
  drive identification codes, 10-2
  drive parameter block, 10-4

FM data encoding, 2-16
format a track, 10-5
head number, 10-5
maximum number of bits to correct on data errors, 10-6
MFM data encoding, 2-16
MMFM data encoding, 2-16
number of heads, 10-6
number of tracks, 10-6
operating voltage, 12-3
read disk status, 10-4
read sectors, 10-5
reset software parameters, 10-4
return current drive parameters, 10-5
sector number, 10-5
start cylinder for reduced write current, 10-6
start cylinder for write precompensation, 10-6
time out for disk diagnostics, 10-6
time out for format, 10-6
time out value for read/write, 10-6
track number, 10-5
verify sectors, 10-5
Floppy, 1-3, 2-7, 2-15
data transfer, 2-16
data transfer rate, 2-7
read/write heads, 2-15
stepping motor, 2-15
theory of operation, 2-15
track 00 detection, 2-15
Winchester, 1-3, 2-7, 2-16
data storage capacities, 2-16
data transfer rate, 2-7
read/write heads, 2-17
stepping motor, 2-16
track 000 detection, 2-16
write sectors, 10-5
Display buffer, see Optional video interface
Divide by zero, 7-1
DMA control, 13-6, 13-9, 14-15
16-bit DMA control, 14-19
8-bit DMA control, 14-17
data transfer, 14-15

DMA controller, 2-5, 14-2, 14-15
operating modes, 14-17
DMA cycle, 14-2
DOS, see Operating systems
see also MS-DOS
DRAM, see Memory
Drive identifier codes, 5-6
Drive type identification, see Winchester drive types

E
Edge connectors, see Backplane
EPROM, see Memory
Equipment configuration, 7-3
Erase cores, 2-17
Erase heads, 2-11
Error Messages, see Firmware
Expansion memory, see Setup program
Extended Graphics Card, see Options

F
Fast mode, 13-15
Firmware, 2-5, 3-1, 3-3
see also Memory
error messages, 3-3
monitor program, 3-1
self tests, 3-3
Floppy disk drive, see Disk drives
Floppy disk drive data transfer, see Disk drives
Floppy disk drive interface cable, 5-14, 5-20
Floppy disk drive read/write heads, see Disk drives
Floppy disk drive stepping motor, see Disk drives
Floppy disk drive Track 00 detection, see Disk drives
Floppy disk drives, 2-15
see also Data storage devices
see also Disk drives
Floppy drive 0, see Setup program
Floppy drive 1, see Setup program
Flux inversion, 2-12, 2-13
FM data encoding, see Data encoding
Function keys, see Keyboard

G
Graphics mode video, see Optional video interface
Graphics tablet, see Peripherals

H
Hardware
    overview, 1-1, 2-1
Hardware jumpers, 5-9
    CPU/RAM card, 5-9
    high density (1.2M) floppy disk drive, 5-13
    I/O card, 5-11
    standard density (360K) floppy disk drive, 5-20
    Winchester disk drives (20M and 40), 5-22
Heads, 2-11
        see also Erase heads
        see read/write heads
Hexadecimal codes, see Keyboard
High resolution color monitor, see Peripherals

I
I/O card, 1-3, 2-5, 14-1
    address decode logic, 14-2
    LEDs, see Diagnostic LEDs
    reset, 14-10
    specifications, see Specifications, 14-1
Increasing the keyboard buffer size, see Keyboard
Input/output
    disk drive port addresses, 10-2
    parallel format, 9-8
    serial format, 9-8
Input/Output card, see I/O card
Instruction cycle, 13-7
Instruction set, 13-2
Intel iAPX 286 Hardware Reference Manual, see Recommended publications
Intel iAPX 286 Programmers Reference Manual, see Recommended publications
Intel, see 80286 Microprocessor
Intel, see 80287 Numeric processor extension
Intel, see Recommended publications

Interrupt

- 00H, 6-7, 6-8, 7-1
- 01H, 6-7, 6-8, 7-2
- 02H, 6-7, 6-8, 7-2
- 03H, 6-7, 6-8, 7-2
- 04H, 6-7, 6-8, 7-2
- 05H, 6-7, 6-9, 9-1
- 08, 6-10
- 08H, 6-7, 6-8, 7-3
- 09, 6-10
- 09H, 6-7, 6-8, 8-1
- 0A, 6-10
- 0AH, 6-7, 6-8, 7-3
- 0B, 6-10
- 0BH, 6-7, 6-9, 9-1, 9-2
- 0C, 6-10
- 0CH, 6-7, 6-9, 9-1, 9-2
- 0D, 6-10
- 0DH, 6-7, 6-9, 9-1, 9-2
- 0E, 6-10
- 0EH, 6-7, 6-9, 10-1, 10-1
- 0F, 6-10
- 0FH, 6-7, 6-9, 9-1, 9-2
- 10H, 6-7, 6-9, 11-1, 11-2
- 11H, 6-7, 6-8, 7-3
- 12H, 6-7, 6-8, 7-5
- 13H, 6-7, 6-9, 10-1, 10-1
- 14H, 6-7, 6-9, 9-1, 9-2
- 15H, 6-7, 6-8, 7-5
- 16H, 6-7, 6-8, 8-1
- 17H, 6-7, 6-9, 9-1, 9-6
- 18H, 6-7, 6-9, 9-1, 9-7
- 19H, 6-7, 6-9, 10-1, 10-11
- 1AH, 6-7, 6-8, 7-11
- 1BH, 6-7, 6-8, 8-1, 8-3
- 1CH, 6-7, 6-8, 7-12
- 1DH, 6-7, 6-9, 11-1, 11-11
- 1EH, 6-7, 6-9, 10-1, 10-12
- 1FH, 6-7, 6-9, 11-1, 11-12
- 70, 6-10
- 70H, 6-7, 6-8, 7-13
- 71, 6-10

71H, 6-7, 6-8, 7-13
72, 6-10
73, 6-10
74, 6-10
75, 6-10
75H, 6-7, 6-8, 7-13
76, 6-10
76H, 6-7, 6-9, 10-1, 10-13
77, 6-10
80287 non-maskable interrupt, 7-13
80287 nonmaskable interrupt, 6-7, 6-8
booting an operating system, 6-7, 6-9, 10-1, 10-11
breakpoint, 6-7, 6-8, 7-2
COM1, 6-7, 6-9, 6-10, 9-1, 9-2
COM2, 6-7, 6-9, 6-10, 9-1, 9-2
communications, 6-7, 6-9, 6-10, 9-1, 9-2
defining characters, 6-7, 6-9, 11-1, 11-12
device control, 6-7, 6-8, 7-5
disk input/output, 6-7, 6-9, 10-1, 10-1
disk parameters, 6-7, 6-9, 10-1, 10-12
divide by zero, 6-7, 6-8, 7-1
equipment configuration, 6-7, 6-8, 7-3
floppy disk controller, 6-10
floppy disk drive, 6-7, 6-9, 10-1, 10-1
how to modify, 6-6
how to use, 6-5
key pressed, 6-7, 6-8, 6-10, 8-1
keyboard break, 6-7, 6-8, 8-1, 8-3
keyboard input/output, 6-7, 6-8, 8-1
LPT1, 6-7, 6-9, 6-10, 9-1, 9-2
LPT2, 6-7, 6-9, 6-10, 9-1, 9-2
memory size, 6-7, 6-8, 7-5
NMI, 6-7, 6-8, 7-2
NMI for 80287, 6-10
nonmaskable, 7-2
nonmaskable interrupt, 6-7, 6-8
overflow, 6-7, 6-8, 7-2
parallel printer, 6-7, 6-9, 6-10, 9-1, 9-2
parallel/serial configuration, 6-7, 6-9, 9-1, 9-7
print screen, 6-7, 6-9, 9-1
printer input/output, 6-7, 6-9, 9-1, 9-6
real-time clock, 6-7, 6-8, 6-10, 7-13

requests, 6-10
rigid drive controller, 6-7, 6-9, 6-10, 10-1, 10-13
second 8259 programmable interrupt controller, 6-7, 6-8, 6-10, 7-3
serial input/output, 6-7, 6-9, 9-1, 9-2
set/read time-of-day, 6-7, 6-8, 7-11
single step, 6-7, 6-8, 7-2
software redirected to IRQ2, 6-10, 7-13
time-of-day, 6-7, 6-8, 6-10, 7-3
timer tick, 6-7, 6-8, 7-12
video initialization, 6-7, 6-9, 11-1, 11-11
video input/output, 6-7, 6-9, 11-1, 11-2
Interrupt requests, 2-5, 2-6, 13-10
see also Keyboard
IRQ0–IRQ7, 14-11
IRQ8–IRQ15, 14-11
maskable, 13-10, 14-10
non-maskable, 13-10
parallel port, 14-6
serial port, 14-5
Interrupts
CPU, 6-8
data storage, 6-9, 10-1
full list of, 6-7
hardware, 6-10
I/O, 6-9, 9-1
keyboard, 6-8, 8-1
parallel port, 9-6
priority of, 6-10
unexpected results, 6-2
video, 6-9, 11-1
Interval timer, see Programmable interval timer
Introduction, see Part I
see also Preface

J
Jump vectors
locations, 6-1
monitor program, 6-1
power-up reset, 6-1
Jumpers, see Hardware jumpers

K
Key arrangement, see Keyboard
Key closure code, see Keyboard
Key matrix, see Keyboard
Key opening code, see Keyboard
Keyboard, 2-6, 2-22, 15-1
  8049 Microcontroller, 15-1
  ASCII codes, 2-22, 15-4
  buffer, 15-10
  cable, 2-22
  calculator keypad, 15-12
  common control keys, 15-8
  connections, see Connections, 2-22
  control Keys, 15-11
  controller, see Controllers
  function keys, 2-22
  hexadecimal codes, 2-22
  increasing the keyboard buffer size, 15-13
  key arrangement, 15-4
  key closure code, 15-4
  key layout, 2-22
  key matrix, 15-1
  key opening code, 15-4
  interface, 14-21
    data transmission, 14-22
    interrupt request, 14-22
  processor, see 8042 Universal peripheral interface
  specifications, see Specifications
  LEDs, 15-4
  nonalphabetic keys, 15-6
  power interface, 15-1
  programming, 8-1
  special function keys, 15-9
  special keys, 15-12
  system code, 15-4
Keyboard-disable lock, 4-2

L
LEDs, see Diagnostic LEDs
    see also Keyboard
Legend of scematic symbols, see Appendix A
Line potential, 12-2

Line select switch, 12-2
Lithium battery, 2-3, 2-6, 12-8
Lock, see Keyboard-disable lock

M

Maskable interrupt requests, see Interrupt requests
MC146818A Real-time clock, see Real-time clock interface
Memory
  access, 13-24
  address format, 7-15
  base memory, 2-4
    see also Setup program
  byte data access, 13-24
  bontrol logic, 13-22
  DRAM, 2-4
  EPROM, 13-20, 13-22
    see also CPU/RAM Card
  expansion memory, 2-4
    see also Setup program
  firmware, 2-5, 13-20
  map, 6-2, 13-21
  monitor ROMs, 13-20
  organization, 13-20
  programming, 7-14
  RAM, 13-20
    see also CPU/RAM Card
  refresh, 2-5, 13-22, 13-25
  ROM, 2-5
  scratchpad RAM, 13-20, 13-22
  typical operations, 7-14
  word data access, 13-25
Memory address bus, 13-20, 13-22
Memory control bus, 13-24
Memory map, see Memory
Memory size, 7-5
Messages, see Error messages
MFM data encoding, see Data encoding
MFM Floppy disk drive data encoding, see Disk drives
MFM-200
    see also Firmware
  description of functions, 6-1
  operation, 6-2

Microprocessor, see 80286 Microprocessor
MMFM data encoding, see Data encoding
Modem, see peripherals
Monitor program, see Firmware
Monitor prompt, 3-2
Monitor ROMs, see Memory
    see also Firmware
Monochrome composite information, see Optional video interface
mouse, see Peripherals
MS-DOS, see Operating systems

N
Nonalphabetic keys, see Keyboard
Non-maskable interrupt requests, see Interrupt requests
Non-maskable interrupt, 7-2
NS16450 Programmable asynchronous communications controller, 14-4
Numeric processor extension, see 80287 Numeric processor extension, 1-2

O
Operating systems
    booting, 10-11
    MS-DOS, 1-2
        CONFIGUR, 9-7
    Xenix, 1-2
    Z-100 mode DOS, 5-8
Optional video interface, 2-19
    character ROM, 2-20
    display buffer, 2-20
    graphics mode video, 2-19
    monochrome composite information, 2-20
    RGB color information, 2-20
    text mode video, 2-19
Options
    card installation, 4-3
    data storage controller card, 1-3, 2-7
        primary I/O address, 2-7
        secondary I/O address, 2-7
    disk drives, 5-7
    extended graphics card, 1-4
    tape Backup Unit, 1-3, 2-7, 2-17
    Z-409 Video Card, 1-4

Z-419 Bit-Mapped Video Graphics Card, 1-4
Z-419 bit-mapped video graphics card, 5-8
Oscillator, 14-13, 14-23
Overview
backplane, see Backplane
hardware, see Hardware
power supply, see Power Supply
technical, see Part II
Owners manual, see Preface

P
Parallel
input/output format, 9-8
interrupts, 9-6
Parallel and serial port addresses, 5-12
Parallel port, 1-3, 2-5, 14-2, 14-6
connector, 14-7
data transfer, 14-8
interrupts, 9-6
interrupt request, 5-12, 14-6
port A, 14-8, 14-9
port addresses, 14-8
port B, 14-9
port C, 14-9, 14-10
port decoder, 14-8
Parallel printer, see Peripherals
Parity, see CPU/RAM card
Part I, Introduction, 1-1
Part II, Technical Overview, 2-1
Part III, System Programming, 6-1
Part IV, System Hardware, 12-1
Peripherals, 1-4, 2-5
graphics tablet, 2-23
high resolution color monitor, 2-23
modem, 2-23
mouse, 2-23
parallel printer, 2-23
Phosphors, 2-20
Pipeline architecture, 13-7
address unit, 13-8
bus unit, 13-7

execution unit, 13-8
instruction unit, 13-7
Pixels, 2-20
Ports
parallel, see Parallel port
serial, see Serial port
map, 6-3
Power supply, 12-1
outputs, 12-3
overview, 2-2
wiring diagram, 12-5
Power switch, 12-3
Powerup self tests, see Self tests
Precompensation timing, see Write precompensation
Preface, iii
PREP, 5-7
Print screen, 9-1, 15-11
Processor
80286, see 80286 Microprocessor
80287, see 80287 Numeric Processor Extension
system Control, see System Control Processor
Program counter, 13-8
Programmable interrupt controller, 13-10, 14-2
Programmable interval timer, 14-12
CPU program instructions, 14-15
modes of operation, 14-13
Programmable timers, see Programmable interval timers
Protected address mode, 13-2, 13-11
Publications
recommended, 1-5

R

RAM (Random access memory), see CPU/RAM card
see also Memory
Raster-scan CRT, see CRT/Monitor operation
Read operation, 2-12, 2-14
Read/write heads, 2-11, 2-17
Real address mode, 13-2, 13-11, 13-24
Real-time clock, 2-6, 14-22
I/O port address, 14-23
Recommended publications, 1-5

Refresh, 2-5, 13-9, 13-22, 14-15
see also Memory
Refresh cycle
arbitration logic, 13-25
Reset, 3-2
RGB color information, see Optional video interface
Rigid disk drive
drive identification codes, 10-2
format a track, 10-5
port addresses, 10-2
ROM (Read-only memory), see CPU/RAM card
ROM, see Memory

S
Schematic illustrations, see Appendix A
Schematic notes, see Appendix A
Scratchpad RAM, see Memory
Second 8259 programmable interrupt controller, 7-3
Segment address, 7-15
Self tests, 3-3
see also Firmware
Powerup, 2-5, 2-6, 14-2
Serial
input/output format, 9-8
Serial input/output
line control status register AH, 9-5
modem control status register, 9-5
Serial port, 1-3, 2-5, 14-2, 14-3
baud rate selection, 9-4
connector, 14-4
data transmission, 14-5
I/O addresses, 14-3
initializing, 9-3
interrupt request, 14-5
parity selection, 9-3
read communications status, 9-4
receive character from, 9-4
send character to, 9-4
word length selection, 9-3
Serial port interrupt request, 5-12
Set/read time-of-day, 7-11

SETUP
  description of functions, 6-1
Setup menu, see Setup program
Setup program, 3-1, 5-1
  base memory, see also Memory, 5-4
  boot drive, 5-4
  date, 5-3
  DST (Daylight Savings Time), 5-3
  expansion memory, see also Memory, 5-4
  floppy drive 0, 5-4
  floppy drive 1, 5-4
  time, 5-2
  video display, 5-5
  video refresh rate, 5-5
  Winchester disk drives, 5-6
Setup, see Setup program
Single step, 7-2
Slow mode, 13-15
Slow mode program execution, 5-8
Software breakpoint, 7-2
Sound
  programming, 7-13
Special function keys, see Keyboard
Special keys, see Keyboard
Specifications
  backplane, 12-6
  I/O Card, 14-1
  keyboard, 15-1
  power supply, 12-1
State generator, see State machine
State machine, 13-4, 13-14, 13-15, 14-17, 14-19
Step pulses, 2-15
Stepper motor, 2-15, 2-17
Supply, see Power supply
Switch
  line select, 12-2
  power, 12-3
System code, see Keyboard
System configuration, 5-1
    see also Hardware jumpers
    see also Setup program
    see also Slow mode

System Control Processor, 2-6, 2-22, 15-1, 15-4
System memory map, see Memory
System memory, see Memory
System, see operating system

T
Tape backup unit, 1-3
        see also Data storage devices
        see also Options
    theory of operation, 2-17
Technical Overview, see Part II
Terminating resistor pack, 5-14, 5-20, 5-22
Tests
    powerup, see Self tests, 2-5
Text mode video, see Optional video interface
Time delays, see Programmable interval timer
Timer
    tick, 7-12
    time-of-day, 7-3
Timers, see Programmable interval timers
Transmission
    Data, see Data transmission
Tunnel erasing, 2-12

V
Video
    alternate select, 11-10
    charater generator routine, 11-8
    compatibility mode, 11-11
    custom graphics mode character set, 11-9
    custom text mode character set, 11-8
    dumb terminal mode, 11-6
    interrupts, 11-2
    normal monochrome character set, 11-8
    read character and attribute, 11-5
    read cursor position, 11-4
    read light pen position, 11-4
    read pixel, 11-6
    report status, 11-7
    scroll area of screen down, 11-5
    scroll area of screen up, 11-5
    select active display page, 11-4

select cursor position, 11-4
select cursor type, 11-4
select palette, 11-5
select scrolling mode, 11-7
set interlace mode, 11-11
set mode, 11-3
set palette registers, 11-7
set pixel, 11-6
set scroll mode, 11-10
standard character set, 11-8
write character and attribute to screen, 11-5
write character only to screen, 11-5
write string, 11-10
Video card, see Options
Video display, see Setup program
Video refresh rate, see Setup program

W
Winchester data storage capacities, see Disk drives
Winchester disk drive
flag bad track, 10-6
Format drive starting at specified track, 10-6
initialize drive characteristics, 10-6
return current drive parameters, 10-6
interfce cable, 5-22
read/write heads, 2-17
stepping motor
track 000 detection
theory of operation, 2-17
Winchester drive type identification, see Winchester drive types
Winchester drive types, 5-6
Wiring diagram, see Power supply
Write cycle, 2-12
Write operation Timing, 2-13
Write operations, 2-10
Write precompensation, 2-14

X
Xenix, see Operating systems

Z

Z-100 compatible graphics, see Z-419 Bit-Mapped Video Graphics Card
Z-100 mode DOS, see Operating systems
Z-100 mode, 5-8
Z-200 PC Series Computer Owners' Manual, see Recommended publications
Z-409 Video Card, see Options
Z-419 Bit-Mapped Video Graphics Card, see Options